Institute for
Naval Oceanography

SP-5
MARCH 1991

A USER'S MANUAL FOR THE PRINCETON NUMERICAL OCEAN MODEL

WILLIAM P. O'CONNOR
INSTITUTE FOR NAVAL OCEANOGRAPHY

The Institute for Naval Oceanography (INO) is operated by the University Corporation for Atmospheric Research (UCAR) under sponsorship of the Naval Oceanographic and Atmospheric Research Laboratory (NOARL). Any opinions, findings, and conclusions or recommendation expressed in this publication are those of the author(s) and do not necessarily reflect the views of NOARL.

ACKNOWLEDGMENTS

This report aids in documenting the primitive equation numerical ocean model originally developed at Princeton University and Dynalysis of Princeton by Drs. George L. Mellor, Alan F. Blumberg, Lakshmi H. Kantha, H. James Herring, Lie-Yauw Oey, Boris Galperin, and others. The model code has evolved slightly as it has been used by different institutions for their purposes and optimized for their computers. This guide documents the version currently in use at the Institute for Naval Oceanography (INO), and may differ slightly from other versions. Thanks are due to Dr. George L. Mellor and especially to Dr. Lakshmi H. Kantha for their explanations of the model. The proofreading was done by Ms. Lydia Harper and Ms. Evelyn Lott.

CONTENTS

1. INTRODUCTION

This is a user's guide for the numerical ocean model developed at Princeton University and presently in use at the Institute for Naval Oceanography. The basic model has been applied successfully to study such diverse regions as Chesapeake Bay (Blumberg 1977), the South Atlantic Bight (Blumberg and Mellor 1983), the Mid Atlantic Bight (Blumberg and Kantha 1985), the Gulf of Mexico (Blumberg and Mellor 1985), New York Harbor (Oey et al 1985a,b,c), Delaware Bay (Galperin and Mellor 1990a,b), and the western Atlantic including the Gulf Stream (Mellor and Ezer 1991). It can model regions from the size of bays and estuaries, to basin scale North Atlantic domains. A good discussion of modeling bays and coastal oceans is given by Blumberg and Oey (1985).

The model is quite developed in its representation of physical processes. The main model characteristics are as follows:

- Primitive equations
- Fully three–dimensional
- Nonlinear
- Flux form of equations
- Boussinesq and hydrostatic
- Terrain following vertical coordinate (σ–coordinate)
- Generalize orthogonal coordinates
- Baroclinic mode
- Free upper surface with barotropic mode
- Turbulence model for vertical mixing
- Smagorinsky horizontal diffusion
- Leapfrog (centered in space and time) time step
- Implicit time scheme for vertical mixing
- Arakawa–C staggard grid

The model is described in several references. The most complete description of the model equations and finite differencing is given by Blumberg and Mellor (1987). A description of the diffusivities in the σ–coordinate system is given by Mellor and Blumberg (1985). The description of the present model cast in generalized orthogonal curvilinear coordinates is given by Blumberg and Herring (1987). A documentation for the use of the model has been produced by Mellor (1990). This user's guide will not attempt to duplicate the above model

descriptions. Rather, it will compliment them by concentrating on how to set parameters in the code for specific applications. The user should first become familiar with the above references, and then use this guide as a supplement when applying the model to a given problem.

The model is configured as a MAIN program and about a dozen subroutines. Generally, only the MAIN program and subroutine BCOND containing the boundary conditions, need to be changed when setting up the model for a specific application. The remainder of the subroutines perform physical calculations common to all applications of the model, such as calculating density and horizontal and vertical advections. This guide will discuss the symbols, arrays, and constants in the MAIN program. The code in the MAIN program is then described, along with the application of the boundary conditions in subroutine BCOND. The use and application of the utilities subroutines to print out arrays of data for easy visual inspection are explained. Then follows a detailed discussion of the necessary input that the user must specify to initialize the model for a specific application. Finally, a specific example of wind forcing in a rectangular domain of constant depth is given. The model input is specified and the resulting output is shown.

As a final note, in the following chapters the information presented is sometimes repetitious, and this is partly a consequence of writing a technical document. However it does save the user the trouble of constantly refering to other parts of the document when trying to find some specific explanation.

2. SYMBOLS, CONSTANTS, AND COMMON BLOCKS

As the model has evolved and been applied to various problems, constants and variables sometimes have been added to or deleted from the program. However, this section describes the constants and variables of the COMMON block and MAIN program that are basic to all applications of the model.

The variables and arrays in the COMMON block are grouped into several labeled common subgroups:

COMMON/BLKCON/	Listing of constants.
COMMON/BLK1D/	Listing of one dimensional arrays.
COMMON/BLK2D/	Listing of two dimensional arrays.
COMMON/BLD3D/	Listing of three dimensional arrays.
COMMON/BDRY/	Listing of boundary value arrays.

In model application, the labeled commons are often put into separate computer files, and then referred to in the MAIN program and subroutines with INCLUDE statements. This facilitates any changes that might be made in the subsequent application. This chapter describes the constants and variables used in the COMMON block and MAIN program.

INDICES

I	Horizontal grid index in X direction, I=1,...,IM.
J	Horizontal grid index in Y direction, J=1,...,JM.
K	Vertical grid index for the sigma (nondimensional) coordinate system. K=1,...,KB. K=1 at the top where Z=0, and K=KB at the bottom where Z=−1.

The parameters specified in the common block are

IM	Maximum number of points in X direction.
JM	Maximum number of points in Y direction.

KB	Maximum number of points in Z direction.

The other parameters are straightforwardly derived from these and are used for dimensioning arrays.

CONSTANTS IN COMMON/BLKCON/

DTE	The external mode (barotropic) time step in seconds.
DTI	The internal mode (baroclinic) time step in seconds.
GRAV	The acceleration of gravity (=9.807 $m\ s^{-2}$).
IINT	The internal mode time step index (IINT=1,...,IEND). Some earlier versions of the model used INT.
IPRINT	The interval (in number of internal mode time steps) at which variables are printed.
TIME	The time in days and fractions of a day.
TPRNU	Weighting factor used in subroutine ADVT to calculate diffusive fluxes over grid spaces (=1.0).
RAMP	The fraction of an inertial period (or sometimes the fraction of one day) that has elapsed since the start of the calculation. If RAMP $\geq$ 1.0, then RAMP = 1.0. It is used to spin up the forcing gradually over one inertial period to minimize spurious wave generation.
UMOL	Background diffusivity (=1.0E-4).

ONE DIMENSIONAL ARRAYS IN COMMON/BLK1D/

Z(KB)	The nondimensional depths in the sigma coordinate system are the levels at which the variables are evaluated. They vary from Z(1)=0 at the surface to Z(KB)= 1.0 at the bottom.

DZ(KB)	The nondimensional grid spacing in the vertical between the Z(K) levels. The last element is set equal to zero DZ(KB)=0.
ZZ(KB)	The sigma levels midway between the Z(K) levels. ZZ(K)=0.5*(Z(K)+Z(K+1))
DZZ(KB)	The grid spacing between the ZZ(K) levels. The last element is set equal to zero DZZ(KB)=0.
DZR(KB)	The inverse of the grid spacing. DZR(K)=1.0/DZ(K). The last element is set equal to zero DZR(KB)=0.

TWO DIMENSIONAL ARRAYS IN COMMON/BLK2D/

AAM2D(IM,JM)	The vertically integrated horizontal eddy viscosity term.
ALAT(IM,JM)	The latitude of a grid box centered on a depth H point.
ALON(IM,JM)	The longitude of a grid box centered on a depth H point.
ANG(IM,JM)	The angle between the two curvilinear axes for a grid box. It is $\pi/2$ for cartesian and spherical coordinates.
ART(IM,JM)	The area of a grid box centered at a depth point. ART(I,J)=DX(I,J)*DY(I,J)
ARU(IM,JM)	The area of a grid box centered at a U velocity point.
ARV(IM,JM)	The area of a grid box centered at a V velocity point.
CBC(IM,JM)	The coefficient of bottom friction. It may be calculated in MAIN and depends on depth and σ-level distribution but is at least set to a minimum value CBCMIN=0.0025.
COR(IM,JM)	The value of the coriolis parameter for each grid box centered on an H point. Earlier versions of the model use COR4(IM,JM), the value of the coriolis parameter divided by 4.

CURV2D(IM,JM)	Initially this array is set equal to COR. In subroutine ADVU this array is calculated to be the sum of COR and the vertical integral of the array CURV resulting from curvature terms in orthogonal curvilinear coordinates. Earlier versions of the model use CURV42D, the above value divided by 4.
D(IM,JM)	The instantaneous depth in meters of the water column in the external mode time step. D(I,J) = H(I,J)+EL(I,J)
DT(IM,JM)	The instantaneous depth of the water column in meters at each internal mode time step. DT(I,J) = H(I,J)+ET(I,J)
DUM(IM,JM)	The U velocity depth mask. It is set equal to one for ocean grid boxes and equal to zero for land grid boxes and those ocean grid boxes with land immediately to the left (lesser I). Then U(I,J)=U(I,J)*DUM(I,J).
DVM(IM,JM)	The V velocity depth mask. It is set equal to one for ocean grid boxes and equal to zero for land grid boxes and those ocean grid boxes with land immediately to the south (lesser J). Then V(I,J)=V(I,J)*DVM(I,J).
DX(IM,JM)	The grid spacing in the X direction for each grid box, in meters.
DY(IM,JM)	The grid spacing in the Y direction for each grid box, in meters.
EGB(IM,JM)	The value of EGF at the previous time step.
EGF(IM,JM)	The external mode sea level in meters averaged over the number of external mode time steps in the internal mode time step. It is calculated by increments using EGF(I,J)= EGF(I,J)+EL(I,J)*ISPI.
EL(IM,JM)	The surface elevation, in meters, as used in the external mode at the central time step.
ELB(IM,JM)	as above, but at previous (back) time step.
ELF(IM,JM)	as above, but at next (forward) time step.

ET(IM,JM)	The surface elevation in meters used in the internal mode at the central time step.
ETB(IM,JM)	as above, but at previous (back) time step.
ETF(IM,JM)	as above, but at next (forward) time step.
FLUXUA(IM,JM)	Array used to calculate terms in the horizontal advection arrays ADVUA and ADVVA in SUBROUTINE ADVAVE. Also used in calculating barotropic free surface by the continuity equation in MAIN.
FLUXVA(IM,JM)	as above.
FSM(IM,JM)	The sea level (and temperature, salinity, density, and vertical velocity) mask. It is set equal to zero for land grid squares and one for ocean grid squares. D(I,J)=D(I,J)*FSM(I,J)
H(IM,JM)	The bottom depth (ocean depth at rest) in meters for each grid box. Land values are typically set to a value between 0 and 1.
PSI(IM,JM)	The stream function, computed diagnostically.
TPS(IM,JM)	Temporary Storage Space array. A number of different variables are stored in this space at various places in the program.
UA(IM,JM)	The vertically averaged (barotropic) velocity in the X-direction used in the external mode calculations, at the central time step, in ms^{-1}.
UAB(IM,JM)	as above, but at the previous (back) time step.
UAF(IM,JM)	as above, but at the next (forward) time step.
VA(IM,JM)	The vertically averaged (barotropic) velocity in the Y-direction used in the external mode calculations, at the central time step, in ms^{-1}.
VAB(IM,JM)	as above, but at the previous (back) time step.
VAF(IM,JM)	as above, but at the next (forward) time step.

WSSURF(IM,JM) The salinity flux at the surface.

WTSURF(IM,JM) The heat flux at the surface.

WUBOT(IM,JM) The X-direction momentum flux at the bottom. It is calculated in subroutine ADVAVE for the external mode, and in subroutine PROFU for the internal mode, with a quadratic resistance law.

WVBOT(IM,JM) The Y-direction momentum flux at the bottom. It is calculated in subroutine ADVAVE for the external mode, and in subroutine PROFV for the internal mode, with a quadratic resistance law.

WUSURF(IM,JM) The X-direction momentum flux at the surface. It is the negative of the surface wind stress divided by the water density, and so it is negative for westerly winds. The wind stress is in Nt m^{-2} and the constant water density 1024 Kg m^{-3} is used, so that WUSURF has dimensions $m^2 s^{-2}$ and is of typical magnitude 10^{-4}.

WVSURF(IM,JM) The Y-direction momentum flux at the surface. It is the negative of the surface wind stress divided by the water density, as above.

THREE DIMENSIONAL ARRAYS IN COMMON/BLK3D/

A(IM,JM,KB) An array used as a holding space by use of an EQUIVALENCE statement in various subroutines. This array is also used directly for calculations in the various vertical profile subroutines, PROFU, etc.

AAM(IM,JM,KB) Horizontal kinematic viscosity. The array is initialized with typical values of 50 - 2000 $m^2 s^{-1}$.

C(IM,JM,KB) An array used as a holding space by use of an EQUIVALENCE statement in various subroutines. This array is also used directly for calculations in the various vertical profile subroutines, PROFU, etc.

DTEF(IM,JM,KB)	An array used in subroutine PROFQ to calculate vertical profiles.
KH(IM,JM,KB)	Vertical diffusivity mixing coefficient for temperature and salinity (real variable).
KM(IM,JM,KB)	Vertical kinematic viscosity momentum mixing coefficient (real variable).
KQ(IM,JM,KB)	Vertical mixing coefficient for turbulence (real variable).
L(IM,JM,KB)	Turbulent length scale (real variable).
Q2(IM,JM,KB)	Twice the turbulent kinetic energy.
Q2B(IM,JM,KB)	as above, but at previous (back) time step.
Q2L(IM,JM,KB)	The product of twice the turbulent kinetic energy times the turbulence length scale, Q2L(I,J,K)=Q2(I,J,K)*L(I,J,K).
Q2LB(IM,JM,KB)	as above, but at previous (back) time step.
RHO(IM,JM,KB)	Density divided by 1000 with 1.025 subtracted. This is done to gain precision since only the gradient of density enters into the calculations. Density calculations are made in subroutine DENS. To obtain the value in $Kg\ m^{-3}$ from the value computed in DENS, add 1.025 and then multiply the result by 1000.
RHOF(IM,JM,KB)	Density at forward time step, defined as above, used in some earlier versions of the model.
RMEAN(IM,JM,KB)	Area averaged density, defined as above.
S(IM,JM,KB)	Salinity in parts per thousand (ppt) with 35 ppt subtracted from real value.
SB(IM,JM,KB)	as above, but at previous (back) time step.
SMEAN(IM,JM,KB)	The average salinity, as above.
T(IM,JM,KB)	Celsius Temperature, with 10 C^o subtracted.

TB(IM,JM,KB)	as above, but at previous (back) time step.
TMEAN(IM,JM,KB)	The average Celsius temperature, as above.
U(IM,JM,KB)	The horizontal velocity in the X-direction, in ms^{-1}.
UB(IM,JM,KB)	as above, but at previous (back) time step.
UF(IM,JM,KB)	as above, but at next (forward) time step.
V(IM,JM,KB)	The horizontal velocity in the Y-direction, in ms^{-1}.
VB(IM,JM,KB)	as above, but at previous (back) time step.
VF(IM,JM,KB)	as above, but at next (forward) time step.
VH(IM,JM,KB)	An array used in subroutines PROFQ, PROFU, and PROFV to calculate vertical profiles.
VHP(IM,JM,KB)	as above.
W(IM,JM,KB)	Vertical velocity in σ-coordinate system.

BOUNDARY VALUE ARRAYS IN COMMON/BDRY/

COVRHE(JM)	Array used in calculating free gravity wave radiation condition on eastern boundary.
COVRHN(IM)	Array used in calculating free gravity wave radiation condition on northern boundary.
ELE(JM)	Sea level in meters on eastern boundary, EL(IM,J).
ELN(IM)	Sea level in meters on northern boundary, EL(I,JM).
ELS(IM)	Sea level in meters on southern boundary, EL(I,1).
SBE(JM,KB)	Salinity on eastern boundary, SB(IM,J,K), with 35 ppt subtracted from real values.

SBN(IM,KB)	Salinity on northern boundary, SB(I,JM,K), as above.
SBS(IM,KB)	Salinity on southern boundary, SB(I,1,K), as above.
TBE(JM,KB)	Temperature on eastern boundary, TB(IM,J,K), with 10 C^o subtracted from real values.
TBN(IM,KB)	Temperature on northern boundary, TB(I,JM,K), as above.
TBS(IM,KB)	Temperature on southern boundary, TB(I,1,K), as above.
UABE(JM)	Barotropic velocity on eastern boundary, UAB(IM,J), in ms^{-1}.
VABN(IM)	Barotropic velocity on northern boundary, VAB(I,JM), as above.
VABS(IM)	Barotropic velocity on southern boundary, VAB(I,1), as above.

OTHER VARIABLES NOT IN COMMON BLOCK

CONSTANTS

AAA	A constant value (typically 500 - 2000 $m^2\ s^{-1}$) sometimes used to initialize the horizontal kinematic viscosity array, AAM(I,J,K)=AAA.
ALPHA	Constant used in weighted averaging the sea level or pressure gradient forcing over the three time steps, in MAIN (typically ALPHA= 0.225).
CBCMIN	Minimum value of the bottom friction term (usually =0.0025).
DAYI	The fraction of one day occupied by one second, or inverse day (=1/86400).

DTE2	Twice the external mode time step (=DTE*2).
DTI2	Twice the internal mode time step (=DTI*2).
FACTOR	The number of internal mode time steps in one day. FACTOR=24*3600/IFIX(DTI)
GSMOD	A function used in some applications to apply a time modulation to an inflow boundary condition in subroutine BCOND. (Gulf Stream modulation)
HORCON	A nondimensional constant used in calculating the horizontal eddy diffusivity (typically 0.01 - 0.10).
IDAYS	The length of time in days that the model simulation will run.
IEND	The total number of internal mode time steps of the model run, IEND=IDAYS*24*3600/IFIX(DTI).
IPRINT	The interval (in number of internal mode time steps) at which variables are printed. It is set by IPRTD1 or IPRTD2, below.
IPRTD1	Time interval in days at which data is written to the standard out. Initially, the model sets IPRINT=IPRTD1*24*3600/IFIX(DTI). Some model versions use the variable IPRNT1.
IPRTD2	Time interval in days at which data is written to the standard out, later in the model run, see ISWTCH, below. Some model versions use IPRNT2.
ISPADV	The frequency interval (in external mode time steps) at which the advection subroutine ADVAVE is called.
ISPI	The ratio of the external to internal time step ISPI=1.0/FLOAT(ISPLIT). It is a real number.
ISP2I	Half the value of ISPI. It is a real number.
ISPLIT	The ratio of the internal to external time steps (=DTI/DTE).

ISWTCH	The number of internal mode time steps after which the model will switch to printing out data at a different interval. It is used in the print section at the end of the MAIN program, in the form IF(IINT.GT.ISWTCH) IPRINT=IPRTD2*24*3600/IFIX(DTI).
MODE	The type of calculation the model performs: MODE=2, performs a two dimensional calculation (bottom stress calculated in ADVAVE). MODE=3, performs a three dimensional calculation (bottom stress calculated in PROFU, PROFV). MODE=4, performs a diagnostic calculation, where the temperature and salinity at each point are held constant in time.
NREAD	The number of data sets read over when reading input from a restart file.
PERIOD	The period of time over which the forcing is ramped or spun up, expressed in days or fractions of a day. It is often the inertial period but is sometimes set equal to one day, or longer as required.
RE	The radius of the earth ($= 6.371 \times 10^6$ m).
SMALL	A small number (1.E–10) added to the denominator of some fractions to insure division by zero does not occur. Used in determining L from Q2LB and Q2B in the initialization.
SMOTH	Constant used in the time filter for the Leapfrog time scheme (typically 0.05 - 0.10).
TIME0	The value of TIME in days at the start of the model calculation (usually TIME0=0.). If the model run is a restart after some time has elapsed, TIME0 will be this elapsed time in days.
VAMAX	The maximum value of VA(I,J) for the domain at the given time step. It is used as a stability check to stop the program if the values become too high.
VAMIN	The minimum value of VA(I,J), as above.

TWO DIMENSIONAL ARRAYS

ADVUA(IM,JM)	Horizontal advection and viscosity in the barotropic equation for UA, calculated in subroutine ADVAVE.
ADVVA(IM,JM)	as above, but for the VA equation.
ADVUU(IM,JM)	Vertical integral of the horizontal dispersion terms used in barotropic UA momentum equation, calculated in SUBROUTINE ADVU.
ADVVV(IM,JM)	As above, but for VA equation, and calculated in SUBROUTINE ADVV.
SSURF(IM,JM)	The surface salinity, used in subroutine ADVT in the calculation of salinity advection. At initialization it is set to SSURF(I,J)=SB(I,J,1).
TRNU(IM,JM)	X-component of the vertical integral of baroclinic pressure gradient (vertical integral of DRHOX), calculated in SUBROUTINE BAROPG.
TRNV(IM,JM)	Y-component, as above.
TSURF(IM,JM)	The surface temperature, used in subroutine ADVT in the calculation of temperature advection. At initialization it is set equal to TSURF(I,J)=TB(I,J,1).
TX(IM,JM)	The X-direction wind stress used in some model applications to read a wind stress in, say dynes cm^{-2} from which WUSURF is obtained.
TY(IM,JM)	The Y-direction wind stress, as above.
UT(IM,JM)	The average value of the barotropic velocity UA multiplied by the depth D, averaged over the internal mode time step.
UTF(IM,JM)	As above, at the forward time step.
VT(IM,JM)	The average value of the barotropic velocity VA multiplied by the depth D, averaged over the internal mode time step.

VTF(IM,JM) As above, at the forward time step.

THREE DIMENSIONAL ARRAYS

CURV(IM,JM,KB) Array that contains the effects of the curvature terms in curvilinear coordinates and the Coriolis effect. It is used in the subroutines ADVU and ADVV. Earlier versions of the model used the array CURV4, the above value divided by 4.

DRHOX(IM,JM,KB) X-component of the internal baroclinic pressure gradient, due to density differences, calculated in SUBROUTINE BAROPG.

DRHOY(IM,JM,KB) as above, Y-component.

PROD(IM,JM,KB) Turbulent kinetic energy production rate.

3. USER SUPPLIED INPUT/INITIALIZATION

The model is very general in both the types of problems for which it can obtain solutions, and the way in which the computer code is applicable to each problem. Generally, it will only be necessary to change the MAIN program and subroutine BCOND to configure the model to a specific problem. Here we shall describe the necessary input the user must supply to the model (a specific example is provided in the chaper describing the box model test case). The model uses the MKS (Meter - Kilogram - Second) system of units. Once some problem dependent values (such as depth H(I,J)) are supplied, the model uses this input to derive other necessary parameters (such as the land mask FSM(I,J)). Other parameters such as HORCON, a constant used in calculating the horizontal eddy diffusivity, may be changed by the user for specific experiments, but are not otherwise considered as user supplied input.

Most variables are initialized to zero at the very start of the MAIN program. This is necessary even if they will be assigned a different value later, because some calculations may exceed the bounds of an array, and take an adjacent value in the COMMON block, before being set to zero in subsequent calculations. If the model is to be spun up from rest, the initial velocities and sea level will be set to zero. If the model is to be restarted after some time, or if an input file has been used to initialize the model, the necessary values are read in with the READ statements at the start of the MAIN program. It would be convenient for the user to append the subroutines DEPTH and DENS to any initialization program. Their use in the model initialization is discussed below.

When initialization files are used to start the model, they usually have been read in using the following conventions.

```
      READ(40) Z,ZZ,DZ,DZZ,ALON,ALAT,DX,DY,H,COR4,ANG,
     1 ART,ARU,ARV,TMEAN,SMEAN,RMEAN,TBN,TBE,TBS,SBN,SBE,SBS,
     2 ELN,ELE,ELS,VABN,UABE,VABS

      READ(50) TIME0,UB,VB,UAB,VAB,TB,SB,ELB,Q2B,Q2LB,KM,KH

      READ(60) WUSUFR,WVSURF,WTSURF
```

The unit 40 contains the physical parameters and climatological data form the problem. The temperature and salinity data is usually taken from the Levitus climatology. Typically, the unit 50 initialization file includes the velocities, temperatures, and salinities created by starting the model from rest and spinning it up over a long enough time to stabilize the variables. The unit 60 contains the surface momentum fluxes (often derived from the Hellerman wind stress climatology) and surface heat flux. The user can modify the initialization files to suit the problem. Since the model uses the centered in time (leapfrog) time step, two levels are necessary to initialize the model. If the model is started from rest this is taken

care of for the velocities when all variables are set to zero at the start of the program. The MAIN program includes the code to set the variables of the central time step equal to those of the back time step (for example, T(I,J,K)=TB(I,J,K)). If both time levels were saved to a restart file, then both can be read in. Restart files are discussed at the end of this chapter.

When the model is started from rest, it is sufficient to specify the variables discussed below. The other variables such as the velocities will have been initialized to zero at the start of the MAIN program.

1. The Problem Dimensions

It is first necessary to specify the size of the domain. These dimensions are

IM	Maximum number of points in the X direction, I=1,...,IM
JM	Maximum number of points in the Y direction, J=1,...,JM
KB	Maximum number of points in the vertical, K=1,...,KB.

These are specified in the PARAMETER statements in the COMMON block. Other parameters such as KBM1=KB-1 are straightforwardly derived and are used for dimensioning arrays. Subroutine DEPTH establishes the vertical resolution with a log distribution at the top and bottom, and a linear distribution in between. This subroutine is called only once when setting up the model domain at the start of the MAIN program. In some applications of the model, the vertical resolution is determined by subroutine DEPTH as part of a prior initialization, and then read as input to MAIN. When subroutine DEPTH is used to determine the vertical spacing, there should be at least six (KB = 6) vertical points for the routine to work properly, and this would be an absolute minimum. The subroutine DEPTH tends to put more points near the surface to resolve the mixed layer. It is the users responsibility to decide how many points are necessary to do this for any specific application. Model applications have been run for the Gulf of Mexico with KB = 12 and 18, and for the North Atlantic Gulf Stream region with KB = 12 and 15. In a call to DEPTH, the subroutine returns the vertical spacing of points in the σ coordinate system. These σ levels vary from Z(1)=0 at the surface to Z(KB)=-1.0 at the bottom. The subroutine also returns the distances between each σ level, DZ(K); the distribution of the midpoints between each level, ZZ(K); and the distance between midpoints of each level, DZZ(K). The last values of the spacing arrays are set equal to zero, DZ(KB)=DZZ(KB)=0. If the user desires to specify the particular vertical spacing, then the arrays Z,DZ,ZZ, and DZZ must be explicitly given.

2. It is necessary to specify the grid spacing in the X and Y directions. This is the grid spacing for each grid box centered on a depth point for the Arakawa-C grid, and the distances are in meters. These are specified as two arrays.

DX(IM,JM)	The grid spacing in the X direction for each grid box, in meters.
DY(IM,JM)	The grid spacing in the Y direction for each grid box, in meters.

The model or an initialization program then uses these values to calculate the areas of the grid boxes.

ANG(IM,JM)	The angle between the two curvilinear axes for a grid box. It is $\pi/2$ for cartesian and spherical coordinates.
ART(IM,JM)	The area of a grid box centered on a depth point
ARU(IM,JM)	The area of a grid box centered on a U velocity point.
ARV(IM,JM)	The area of a grid box centered on a V velocity point.

The areas of the grid squares are made by the calculations

```
ART(I,J)=DX(I,J)*DY(I,J)
ARU(I,J)=0.25*(DX(I,J)+DX(I-1,J))*(DY(I,J)+DY(I-1,J))
ARV(I,J)=0.25*(DX(I,J)+DX(I,J-1))*(DY(I,J)+DY(I,J-1))
```

with the special cases of

```
ARU(1,J)=ARU(2,J)
ARV(1,J)=ARV(2,J)
ARU(I,1)=ARU(I,2)
ARV(I,1)=ARV(I,2).
```

The most common coordinate systems used are cartesian and spherical, but other orthogonal curvilinear coordinate systems can be set up for particular domains using a grid generation program.

3. The value of the coriolis parameter must be specified for each grid box. This is the array COR(IM,JM). This feature allows the model to be used for large ocean domains, while for

smaller domains this value can be the same for each grid box. Some earlier versions of the model use the value of the coriolis parameter divided by 4, COR4(IM,JM), and have the dynamical equations programmed accordingly.

Some versions of the model have the arrays ALAT(IM,JM) and ALON(IM,JM), which give the latitude and longitude of each grid box. These arrays are useful for recalling the location of each grid box for graphics work, and calculating the coriolis parameter, but are not used directly in the model calculations.

4. The bottom topography is specified with the ocean depth in H(IM,JM). For some applications it may be advisable to filter the bottom topography with a Shapiro (1970) filter to remove $2\Delta x$ noise. This depth array is specified over the entire domain, both land and ocean. For the land grid squares, H is set to a value of between 0.0 and 1.0, typically H(I,J)=0.5. The finite difference calculations proceed over all the grid squares of the domain, both land and ocean, and the values calculated at land points are subsequently set to zero with appropriate land and velocity masks. These masks for land and velocity then make use of the criterion that $0.0 \leq H(I,J) \leq 1.0$ to define a land grid square. However, the time step calculations make use of division by H(I,J), and so everywhere H must have a nonzero value. When the model is run in the baroclinic mode with, say, 10 vertical levels (KB=10), then the minimum depth should be several meters, to avoid computational instabilities. This may also depend on the range of depths over the model domain. When the model was used to simulate hurricane landfalls in the Gulf of Mexico with KB = 18, the minimum depth used was 10 m. For barotropic (vertically integrated) mode calculations, the shallowest depth could be about two meters. However, these features are very problem specific, and depend a great deal on the model domain and the forcing.

5. The Temperature, Salinity, and Density

It is necessary to specify the initial and mean temperatures and salinities as input to the model.

TMEAN(IM,JM,KB)	The area averaged temperature in degrees Celsius, with 10 C^o subtracted before input to the model. It is used in the advection subroutine ADVT.
SMEAN(IM,JM,KB)	The area averaged salinity in parts per thousand (ppt), with 35 ppt subtracted before input to the model. It is used in the advection subroutine ADVT.
TB(IM,JM,KB)	The initial temperature in degrees Celsius, with 10 C^o subtracted before input to the model.

SB(IM,JM,KB) The initial salinity in parts per thousand, with 35 ppt subtracted before input to the model.

The modeler should subtract 10 C^o from the temperature and 35 ppt from the salinity as part of the model initialization. This results in additional precision, since for most calculations only the gradients of temperature and salinity are taken. However, in subroutine DENS, these values are added back to the temperature and salinity before the density is calculated. All calculations on the CRAY-YMP are performed using 64 bit precision. When the model is run on a machine with 32 bit precision, the calculations in subroutine DENS should be made in double precision.

The temperature and salinity may be taken from the Levitus climatology or specified appropriately for the problem. Generally, when the model is initialized from a climatology (as opposed to restarting the model after some period of time), the arrays TMEAN and TB will be set to the same initial values, and the arrays SMEAN and SB will be set to the same initial values.

RMEAN(I,J,K) The area averaged density divided by 1000 with 1.025 subtracted. To obtain the value in Kg m^{-3} form the value computed in DENS, add 1.025 and then multiply the result by 1000.

The area averaged density must be supplied to initialize the model. It is initially determined by a call to subroutine DENS using the initial temperature and salinity as input parameters. When an initialization program is used to supply input to the model, this program would have to include the subroutine DENS. The density RHO(I,J,K) is the returned parameter from DENS and then the mean density is then defined by setting RMEAN(I,J,K)=RHO(I,J,K) . The mean density is subtracted from the density at the beginning of subroutine BAROPG, which computes the baroclinic pressure gradient. This results in some increase in accuracy of the calculations. At the end of subroutine BAROPG the mean density is added back to the value of the density. Here we note that subroutine DENS computes the density divided by 1000 with 1.025 subtracted. This is done to gain precision since only the gradient of density enters into the calculations. To obtain the density value in Kg m^{-3} from the value computed in subroutine DENS, add 1.025 and then multiply the result by 1000.

6. The boundary value arrays should be specified. The boundaries and boundary value arrays are dependent on the particular problem. The model version supplied assumes that the western boundary is a land boundary, while the north, south, and east boundaries can be open and have boundary value arrays dimensioned for them. The user can configure the model to different boundary geometries. If one of the north, south, or east boundaries is a

land boundary, the boundary value arrays need not be used. At a land boundary the masking is all that is necessary to determine the correct boundary values there. If the model requires an open western boundary, the user will have to define appropriate boundary value arrays. The sea level, temperature, salinity, and velocity boundary value arrays for the northern, eastern, and southern boundaries are

ELE(JM)	Sea level EL(IM,J) on eastern boundary.
ELN(IM)	Sea level EL(I,JM) on northern boundary.
ELS(IM)	Sea level EL(I,1) on southern boundary.
TBN(IM,KB)	Temperature in C^o on northern boundary, with 10 C^o subtracted from real values.
TBS(IM,KB)	Temperature in C^o on southern boundary, as above.
TBE(JM,KB)	Temperature in C^o on eastern boundary, as above.
SBN(IM,KB)	Salinity in ppt on northern boundary, with 35 ppt subtracted from the real values.
SBS(IM,KB)	Salinity in ppt on southern boundary, as above.
SBE(JM,KB)	Salinity in ppt on eastern boundary, as above.
UABE(JM)	Barotropic velocity UAB(IM,J) on eastern boundary.
VABN(IM)	Barotropic velocity VAB(I,JM) on northern boundary.
VABS(IM)	Barotropic velocity VAB(I,1) on southern boundary.

For purposes of initialization, it is probably best to set these boundary value arrays to the appropriate values of climatology, the same values for TB(I,J,K) with 10 C^o subtracted, and SB(I,J,K) with 35 ppt subtracted. The boundary conditions vary greatly with the problem, and this will be discussed further in the section on subroutine BCOND.

7. The wind stress

The surface momentum flux must be specified. These are the arrays

WUSURF(IM,JM)	The X–direction momentum flux at the surface. It is the negative of the surface wind stress divided by the water density, and so it is negative for westerly winds. The wind stress is in Nt m^{-2} and the constant water density 1024 Kg m^{-3} is used, so that WUSURF has dimensions $m^2 s^{-2}$ and is of typical magnitude 10^{-4}.
WVSURF(IM,JM)	The Y–direction momentum flux at the surface. It is the negative of the surface wind stress divided by the water density, and so it is negative for northerly winds, as above.

In some applications of the model it is convenient to read in the data in the centimeter – gram – second units of dynes cm^{-2}. When this value is divided by the water density 1.0 gm cm^{-3} the result is of magnitude one $cm^2 s^{-2}$. This data is usually read into the model with the arrays.

TX(IM,JM)	The wind stress in the X direction divided by the water density, in units of $cm^2 s^{-2}$.
TY(IM,JM)	The wind stress in the Y direction divided by the water density, in units of $cm^{-2} s^{-2}$.

and the values of WUSURF and WVSURF can be obtained from them by changing the sign and multiplying by the scaling factor 10^{-4}. These arrays TX and TY do not appear in all versions of the model, but can be added by the user.

8. The surface temperature and salinity fluxes

These surface fluxes are presently set to zero in the time integration in the MAIN program. The user can comment this out and supply the input if desired.

WTSURF(IM,JM)	The surface temperature flux.

WSSURF(IM,JM)	The surface salinity flux.

9. The Time Steps

DTE	The external mode time step in seconds.
DTI	The internal mode time step in seconds.
ISPLIT	The ratio of the internal to external mode time steps (=DTI/DTE), expressed as an integer.

The internal (baroclinic) time step DTI and external (barotropic) time step DTE must be set appropriate to the problem. The external time step is the shorter and its value is governed by the CFL computational stability criterion, where this time step must be shorter than the time it takes the free surface gravity wave to travel between any two grid points. The model code includes the calculation of the minimum time step to meet the CFL criterion for each grid square, based on the grid spacing and ocean depth. The resulting data is stored in the array TPS (Temporary Storage Space). In practice, after considering the external mode CFL criterion, set values for DTI and ISPLIT. For example, to obtain an external mode time step of 30 seconds, and an internal mode time step of 600 seconds, set DTI=600.0 and ISPLIT=20, so that the model will subsequently calculate the value DTE=30.0 with the command DTE=DTI/FLOAT(ISPLIT).

10. The TIME and print intervals

The model simulation time is measured in days and fractions of a day, and two parameters must be set to the correct value.

TIME	The elapsed model time in days and fractions of a day.
TIME0	The value of TIME in days at the start of the model calculation. If the model run is started from a restart file, TIME0 will be the time read in from the restart file.

Initially the model is set with TIME=0.0. When information is written to a file at the end of some time interval, the variable TIME is usually the first variable written to the file. If the model is restarted after some time, be sure that the model uses (or reads in from an input file) the appropriate value TIME0. The model then sets TIME=TIME0.

The model is integrated forward in time counting the internal mode time steps, using the loop DO 9000 IINT=1,IEND, and so it is necessary to specify

IEND	Total number of internal mode time steps.

For example, if the model is to be run for ten days, with the internal mode time step DTI=600.0 seconds, then set IEND=1440. In the current version of the model, the user specifies

IDAYS	Time in days of model simulation.

and the model calculates the value of IEND with the statement

IEND = IDAYS*24*3600/IFIX(DTI).

The intervals at which files are to be printed out or saved must be specified. These specifications are based on the number of internal mode time steps needed to cover the time interval. The model comes with two print intervals at which data can be written to the standard out (unit 6)

IPRTD1	Time interval in days at which data is written to the standard out, used at start of model. The model sets IPRINT=IPRTD1*24*3600/IFIX(DTI).
IPRTD2	Time interval in days at which data is written to the standard out, later in the model run.
ISWTCH	The number of internal mode time steps after which the model will switch to printing out data at a different interval. It is used in the printing section at the end of the main program, in the form IF(IINT.GT.ISWTCH) IPRINT=IPRTD2*24*3600/IFIX(DTI).

It is useful to have two print intervals for short debugging runs, but otherwise it is only necessary to use one (set both IPRTD1 and IPRTD2 to the same number of days).

In the version of the model at INO, the intervals at which files are be written can be specified with the variables

ISVEL	Interval at which data is saved to a file.
ISVTS	Interval at which data is saved to a file.

In the version of the model at INO, these values are specified in days and the model then calculates the equivalent number of internal mode time steps, with the command

ISVEL=ISVEL*24*3600/IFIX(DTI).

The exact specifics of how the print and file save intervals are set may depend on the exact version of the model. The user can easily adapt the proceedure to save as many files as necessary at whatever intervals are desired.

11. The MODE or type of calculation to be performed must be specified. It can be one of the three values

MODE=2.	Model performs a two dimensional (X,Y) vertically integrated calculation for the average velocities UA(I,J), VA(I,J), and the sea level EL(I,J).
MODE=3	Model performs a fully three dimensional calculation.
MODE=4	Model performs a three dimensional diagnostic calculation. The temperature and salinity (and hence density) at each grid point are constant, so there is no advection of temperature, salinity, or density.

12. The vertical kinematic viscosity KM, vertical heat diffusivity KH, twice the turbulent kinetic energy Q2B, turbulence length scale L, and Q2LB(I,J,K)=Q2B(I,J,K)*L(I,K,J) can be initialized with zero values or set to some small value on the order 10^{-4}. The horizontal kinematic viscosity AAM(I,J,K) can be set to some value (typically 50 - 2000 $m^2 s^{-1}$) depending on the dynamics of the problem and the grid spacing.

13. The Spin up time or PERIOD over which the model forcing is increased linearly from zero to its actual value should be specified.

PERIOD	The period of time in days over which the forcing is spun up. It is usually set to the inertial period in days, or sometimes one day, but can be set to longer

	times to spin up special problems (such as a model with very shallow depths).
RAMP	The factor that increases linearly from zero to one to spin up the forcing, RAMP=TIME/PERIOD.

The wind stress forcing and baroclinic pressure gradient forcing in the numerical integration of the MAIN program, and the inflow boundary velocities in subroutine BCOND are multiplied by the factor RAMP.

14. For long time simulations of the model, it may be desirable to write model data to a restart file. It is best to write binary data files for greater precision. Since the model uses a centered difference (leapfrog) time step, two time levels are necessary for a 'seemless' restart. This can be done with the command.

```
      WRITE(77) TIME,
     1 CURV42D,WUBOT,WVBOT,AAM2D,UA,UAB,VA,VAB,EL,ELB,
     2 ET,ETB,EGB,UTB,VTB,U,UB,W,V,VB,T,TB,S,SB,RHO,ADVUU,
     3 ADVVV,ADVUA,ADVVA,KM,KH,KQ,L,Q2,Q2B,AAM,Q2L,Q2LB
```

4. The MAIN Program

The MAIN program will vary somewhat depending on which version of the model is being used, and its application to a specific problem, but it consists of two logical parts. The first is the initialization, where the model parameters are prepared for the specific problem. The second logical part is the numerical integration of the model forward in time, which is done by the loop DO 9000 IINT=1,IEND. We shall discuss these parts separately.

The model form will also depend on whether it was optimized for a supercomputer with an array processor. In the latter case two arrays can be set equal with a single statement and it need not be enclosed in a DO loop. Then the statement numbers may vary somewhat with the specific version of the model. The statement numbers in the model initialization will vary with the application, depending on exactly what must be specified. However, the statement numbers required for the processing of the model numerical integration of the MAIN program are standard features and do not differ significantly between versions.

Part 1. The Initialization

This is the part of the model that will vary most for any specific application. The necessary initialization data can be included here, or written to a separate initialization data file and then read into the MAIN program for each model run. This latter method would be the most efficient if the model is to be run many times.

First the COMMON block is given explicitly or called from a separate file with an INCLUDE statement. The MAIN program then declares and dimensions several more arrays and constants. Most of these are physical parameters. A complete listing and description of the arrays and constants in the COMMON block and the other arrays and constants used in the MAIN program is given in Chapter 2, Symbols, Constants, and Common Blocks.

At the start of the MAIN program, most of the arrays are initialized to zero, even though other values will be assigned later in the program. This is necessary because the code was written for efficiency on vector processing computers, where calculations over the domain sometimes exceed the bounds of the array. Values are then taken from adjacent arrays stored in the COMMON block, and so all variables must be initialized to some value at the very beginning to avoid an 'undefined variable' error message. This initialization is done with Fortran DO loops or DATA statements. It is recommended that DO loops be used since initializing large arrays with DATA statements is very inefficient on some machines.

Next, the individual problem specifications are supplied appropriate to the problem. The model reads in the required data to start the model from initialization or restart files, or specifies it in this section of the MAIN program. These are described in detail in the section on User Supplied Input, and some discussion is repeated here. Basically, the time steps and print and file save intervals are specified, and any other model parameters are specified for the physical application.

If the model is to be spun up from rest, the initial velocities and sea level will be set to zero. When initialization files are used to start the model, they usually are read in using the following conventions

```
      READ(40) Z,ZZ,DZ,DZZ,ALON,ALAT,DX,DY,H,COR4,ANG,
     1 ART,ARU,ARV,TMEAN,SMEAN,RMEAN,TBN,TBE,TBS,SBN,SBE,SBS,
     2 ELN,ELE,ELS,VABN,UABE,VABS

      READ(50) TIME0,UB,VB,UAB,VAB,TB,SB,ELB,Q2B,Q2LB,KM,KH

      READ(60) WUSUFR,WVSURF,WTSURF.
```

The unit 40 contains the physical parameters and climatological data form the problem. The temperature and salinity data is usually taken from the Levitus climatology. Typically, the unit 50 initialization file includes the velocities, temperatures, and salinities created by starting the model from rest and spinning it up over a long enough time to stabilize the variables. The unit 60 contains the surface momentum fluxes (often derived from the Hellerman wind stress climatology) and surface heat flux. The user can modify the initialization files to suit the problem. Since the model uses the centered in time (leapfrog) time step, two levels are necessary to initialize the model. If the model is started from rest this is taken care of for the velocities when all variables are set to zero at the start of the program. The MAIN program includes the code to set the variables of the central time step equal to those of the back time step (for example, T(I,J,K)=TB(I,J,K)), and this is discussed below.

For long time simulations of the model, it may be desirable to write model data to a restart file. It is best to write binary data files for greater precision. Since the model uses a centered difference (leapfrog) time step, two time levels are necessary for a 'seemless' restart. The restart files are read over and the values from the last (NREAD–th) one is used to initialize the model with the command.

```
      DO 10 N=1,NREAD
      READ(77) TIME0,
     1 CURV42D,WUBOT,WVBOT,AAM2D,UA,UAB,VA,VAB,EL,ELB,
     2 ET,ETB,EGB,UTB,VTB,U,UB,W,V,VB,T,TB,S,SB,RHO,ADVUU,
     3 ADVVV,ADVUA,ADVVA,KM,KH,KQ,L,Q2,Q2B,AAM,Q2L,Q2LB
    10 CONTINUE
```

The following describes the most common form of the initialization part of the MAIN program. The processes are described with as little reference to Fortran statement numbers as possible.

The eddy diffusivity is initialized to a constant value, typically AAA = 500 - 2000 $m^2 s^{-1}$ with the statement AAM(I,J,K)=AAA.

Any data that is not already in the MKS system should be converted to this system. For example, if the wind stress is in dynes cm^{-2}, it can be multiplied by the factor 1.E–4 to convert it to $m^2 s^{-2}$ which is wind stress divided by water density, TX(I,J)=TX(I,J)*1.E–4.

The surface temperature and salinity arrays (used in heat and salinity flux calculations) are initialized with the statements

TSURF(I,J)=TB(I,J,1)

SSURF(I,J)=SB(I,J,1).

The surface elevation used in the internal mode calculations is initialized with the statement DT(I,J)=H(I,J)+ELB(I,J).

The command TIME=TIME0 sets the TIME variable to the value at the last restart file time.

The inverse of the vertical grid spacing is calculated with the command DZR(K)=1./DZ(K) for use in subsequent calculations. This is now done in subroutine DEPTH, but some earlier versions of DEPTH did not make this calculation.

The variable PERIOD is that time in days or fractions of a day over which the model is spun up with the RAMP variable. It is often calculated as the inertial period for the latitude of the center grid square of the domain but can be a longer period if required by the problem.

The Coriolis parameter is calculated for each grid square of the domain and stored in the array COR(I,J). Previous versions of the model used the value of the Coriolis parameter divided by 4, stored in COR4(I,J). The statement CURV2D(I,J)=COR(I,J) sets the array CURV2D equal to the value of the Coriolis parameter. In the subroutine ADVU the array CURV2D is calculated to be the sum of COR and the vertical integral of a curvature array CURV, resulting from curvilinear coordinate systems.

The land and velocity masks are determined from the bottom topography in three DO loops. We recall that the model grid extends over all land and ocean points. Land grid points are distinguished by having a depth between zero and one ($0 < H(I,J) < 1.0$). They must be assigned some nonzero value because the calculations that include division by the depth take place over the entire grid of land and ocean points. The loop DO 30 J=1,JM determines the land mask FSM(I,J). Using the above depth criterion, this array is assigned the value 0.0 for land grid squares and 1.0 for ocean grid squares. The grid squares for the model are those for the Arakawa–C grid. The depth (and temperature, salinity, vertical velocity, and sea level) points are at the center of the grid box, the U velocity points are the midpoint of the left side, and the V velocity points are at the bottom (south). The 'masking' or setting to zero of some value over land points is usually done in suboutine BCOND by array multiplication of the variable times the mask FSM, eg., EL(I,J)=EL(I,J)*FSM(I,J).

The loop DO 35 J=1,JM determines the V velocity mask DVM(I,J). If the grid square is a land grid or an ocean grid with a land grid immediately to the south (lesser J value), the value is zero. Otherwise, the mask is set equal to 1.0. The velocities are usually masked in subroutine BCOND by array multiplication, eg., VF(I,J,K)=VF(I,J,K)*DVM(I,J).

The U velocity mask DUM(I,J) is determined in the loop DO 40 J=1,JM If the grid square is a land grid or an ocean grid with a land grid immediately to the left, the value is

zero. Otherwise the mask is set equal to 1.0. The velocities are usually masked in subroutine BCOND by array multiplication, eg., UA(I,J)=UA(I,J)*DUM(I,J).

Some versions of the model have the statements DUM(1,J)=0.0 and DVM(1,J)=0.0. They make the entire western boundary a land boundary which was used in some Gulf Stream models to put a wall across the Florida Straits. It is a problem dependent feature.

The call to subroutine VABFIX is used in some models to specify temperatures, salinities, and inflow velocities across an open boundary. This is done with models of the Gulf Stream region to specify an inflow to the domain along the northern boundary across the continental shelf, in order to obtain Gulf Stream separation at Cape Hatteras. This is also done with models of the Gulf of Mexico to set the inflow conditions at the Yucatan Straits. This subroutine is not used in all models.

The initial bottom friction coefficient CBC(I,J) is calculated depending on the depth and grid spacing, but is at least set to the minimum value CBCMIN=0.0025.

The user can print out for inspection whatever initial values are desired. A description of the various subroutines for writing output is discussed in the section on utilities subroutines.

The value of the CFL criterion for each grid square, considering the dimensions of the grid and the ocean depth, is calculated and stored in the Temporary Storage Space array TPS(I,J). The value of the external mode time step DTE must be smaller than the minimum value in this array. Note that the array TPS is used to store other values in other parts of the program.

In some versions of the model the the top and bottom levels of the turbulence length scale are set equal to zero with the statements L(I,J,1)=0.0 and L(I,J,KB)=0.0. Recall that the array L is defined as a real variable. The statements

```
L(I,J,K)=Q2LB(I,J,K)/(Q2B(I,J,K)+SMALL)
KQ(I,J,K)=0.20*L(I,J,K)*SQRT(Q2B(I,J,K))
```

first initialize the turbulence length. Note that Q2B is twice the turbulent kinetic energy. The parameter SMALL=1.E-10 in the denominator ensures that there will be no division by zero, when using the initial value of Q2B that was set to zero. The second line initializes the vertical mixing coefficient for turbulence, which is defined as a real number.

The model uses the leapfrog scheme for integration in time. It is necessary to keep three levels for the variables that are integrated in time, and usually the letter B at the end of a variable name denotes the BACK (or n-1) time step, and the letter F at the end of a variable name denotes the FORWARD (or n+1) time step. However, in order to save memory, the variable UF is used to represent the (n+1) time level for T and Q2, and the variable VF is used to represent the (n+1) time level for S and Q2L. It is necessary to initialize the model by specifying the variables at two time levels, the back and center levels. It is sufficient to set the variable at the center time step equal to the variable at the back time step. This is what is done in the array calculations

```
UA(I,J)=UAB(I,J)
VA(I,J)=VAB(I,J)
EL(I,J)=ELB(I,J)
ETB(I,J)=ELB(I,J)
ET(I,J)=ETB(I,J)
ETF(I,J)=ET(I,J)
D(I,J)=H(I,J)+EL(I,J)
DT(I,J)=H(I,J)+ET(I,J)
Q2(I,J,K)=Q2B(I,J,K)
Q2L(I,J,K)=Q2LB(I,J,K)
T(I,J,K)=TB(I,J,K)
S(I,J,K)=SB(I,J,K)
U(I,J,K)=UB(I,J,K)
V(I,J,K)=VB(I,J,K).
```

If the model is initialized by reading in a restart file that has two time levels, then the above commands should not be used.

The subroutine DENS is called to calculate the density, and the mean density can be initialized to this value, if it has not been defined before, with the command, RMEAN(I,J,K) = RHO(I,J,K).

The subroutine BAROPG is called to obtain the baroclinic pressure gradient CALL BAROPG(DRHOX,DRHOY,TRNU,TRNV). The four parameters in the call statement are calculated in BAROPG and returned. The parameters DRHOX(I,J,K) and DRHOY(I,J,K) are the X and Y components of the internal baroclinic forcing term, resulting from density differences. The parameters TRNU(I,J) and TRNV(I,J) are the components of the vertically integrated pressure gradient forcing. The components DRHOX, DRHOY are used in the three dimensional calculations, and their vertical integrals TRNU and TRNV are used in the two dimensional barotropic calculations.

Part 2. The Integration

The numerical integration takes place in the loop DO 9000 IINT=1,IEND. The index IINT counts the internal mode (longer) time step DTI, making three dimensional (baroclinic mode) calculations. For every one of these internal mode time steps, the model performs ISPLIT external mode (barotropic or two dimensional) calculations, integrating with the external mode (shorter) time step DTE. (Recall that the integer ISPLIT is the ratio of the internal mode to external mode time steps.) This is done in the loop DO 8000 IEXT=1,ISPLIT.

We begin with a discussion of the internal mode time integration. The model first determines the variable TIME, giving its value in days and fractions of a day

```
TIME=DAYI*DTI*FLOAT(IINT)+TIME0.
```

Some versions of the model calculate the Julian date which is useful for modeling case studies with tides or specific wind forcing data sets

```
JDAY=DAYI*DTI*FLOAT(IINT)+JDAY0.
```

The value of RAMP = TIME/PERIOD is between zero and one, and is the fraction of the variable PERIOD (in days) that has elapsed since the start of the time integration. The value of PERIOD could be arbitrarily set to one day (PERIOD=1.) or longer as required to spin up a particular problem. After the time interval PERIOD has elapsed, the value of RAMP is held constant at unity, with the command, IF(RAMP.GT.1.0) RAMP=1.0. This value is used to spin up a forcing gradually and avoid exciting transient waves in the model. Be sure that RAMP is set equal to one if the model is initialized from a restart file.

Generally, all initial forcings to the model should be ramped. The vertically integrated pressure gradient forcing components are ramped where they appear in the barotropic equations of motion, with the terms RAMP*TRNU, and RAMP*TRNV, and if the model is forced by any inflow boundary conditions, they are also ramped in subroutine BCOND.

For the three dimensional calculations (MODE=3 or 4) the model calls subroutine BAROPG with the command

```
IF(MODE.NE.2) THEN
CALL BAROPG
ENDIF
```

to calculate the vertically integrated baroclinic pressure gradient TRNU, TRNV. Then the statement

```
TRNU(I,J)=TRNU(I,J)+ADVUU(I,J)-ADVUA(I,J)
```

adds the vertical integral of the horizontal dispersion terms ADVUU calculated in subroutine ADVU, and subtracts the barotropic horizontal advection ADVUA calculated in subroutine ADVAVE. This is made here for the internal mode calculations.

The model then calculates the horizontal kinematic viscosity coefficient AAM(I,J,K) at each level. For some applications it can be kept at a constant value (typically 500 - 2000 $m^2 s^{-1}$).

The vertically integrated kinematic viscosity is initialized to zero with the statement AAM2D(I,J)=0.0, and the statement

```
AAM2D(I,J)=AAM2D(I,J)+AAM(I,J,K)*DZ(K)
```

vertically integrates the horizontal kinematic viscosity.

The model will calculate an average of the external mode variables over the ISPLIT external mode time steps in the internal mode time step. The variable EGF(I,J) is this average of the external mode sea level EL(I,J). The variables UTF(I,J) and VTF(I,J) are these averages of the external mode velocities, multiplied by the depth D(I,J)=H(I,J)+EL(I,J). This averaging calculation is performed as the time integration takes place. Before the start of the external mode calculation, these averages must be initialized. This is done in the loops for

EGF(I,J)=EL(I,J)*ISPI

UTF(I,J)=UA(I,J)*(D(I,J)+D(I-1,J))*ISP2I.

Recall that ISPI is the (real value) number of external mode time steps in one internal mode time step. The real value ISP2I is half this value. The ISP2I value is used to incorporate the division by 2 when averaging the two values of D(I,J). The averaging at each external mode time step will take place at the end of the external mode calculation.

The External Mode

The loop DO 8000 IEXT=1,ISPLIT, performs the external mode time step integrations. There are ISPLIT external mode (short) time steps DTE in one internal mode (long) time step DTI. The loop DO 405 J=2,JMM1, calculates the vertically averaged velocity fluxes FLUXUA(I,J), FLUXVA(I,J) through the sides of a grid square centered on a depth point. The loop DO 410 J=2,JMM1, calculates the barotropic sea level ELF(I,J) from the continuity equation. The command CALL BCOND(1), applies the boundary conditions and masking to the barotropic sea level ELF(I,J).

The integer ISPADV is the frequency interval in external mode time steps at which the advection subroutine ADVAVE is called. The model is usually set with ISPADV=1, and this is recommended for strongly advective regimes, such as Gulf Stream models. The statement

IF(MOD(IEXT,ISPADV).EQ.0) CALL ADVAVE(ADVUA,ADVVA,MODE)

makes this determination and call. This subroutine returns ADVUA, ADVVA, the nonlinear velocity advections of the vertically integrated velocity.

The barotropic velocity equation for UAF(I,J) is calculated in the loop DO 420 J=2,JMM1, and evaluates the forcings of the Coriolis, sea level pressure gradient, vertically integrated baroclinic pressure gradient due to density differences, and wind stress: The Coriolis terms are multiplied by the array CURV42D(I,J). Terms multiplied by GRAV are the sea level pressure gradient. A weighted average over the three time steps is sometimes taken using the parameter ALPHA (typically, ALPHA=0.225). This is not absolutely necessary, but is done to increase the time step. The vertically integrated baroclinic pressure gradient terms are TRNU. The surface momentum flux (wind stress forcing) terms are WUSURF. The quadratic bottom friction terms WUBOT are calculated in subroutine ADVAVE. The array ARU(I,J) is the area of a grid square centered on the U(I,J) velocity point, and is necessary because the equations are cast in generalized coordinates. The loop DO 425 J=2,JMM1,for the velocity UAF(I,J) integrates the vertically averaged U velocity forward in time using the centered difference (leapfrog) time step.

The loops DO 430 and DO 435 perform the analogous integration of the barotropic V velocity equation. The command CALL BCOND(2), calls subroutine BCOND to apply the boundary conditions and masking to the barotropic velocities UAF and VAF.

A smooth value for ETF(I,J), the surface elevation used for the internal mode calculations, is calculated with the statement

IF(IEXT.LT.(ISPLIT-2)) GO TO 440

and the subsequent commands up to the 440 CONTINUE, based on the last three external mode time steps.

The model next tests for instabilities which might occur if, for example, DTE or DTI is too large. The loop DO 441, in effect, finds the largest absolute value of the elements in the velocity array VAF. If this is greater than some specified value (in this case 100.0 ms^{-1}), the model will GO TO 9001, print out arrays, and then STOP. This allows a normal exit to the program when nonlinear instabilities are causing the model to blow up, and the printed arrays can give some idea of what is happening. Other instability checks can be written based on sea level height.

It is necessary to filter the variables over the three time steps of the central differencing. Otherwise the solutions at the odd and even time steps will tend to diverge. The model performs this filtering at every time step, but a model could be written that does this at every several time steps if desired. The loop DO 500 J=1,JM does this time filtering and then resets the time sequence for the next external mode time step calculation.

The ongoing averaging of the external mode variables over the ISPLIT external mode time steps is performed in the loop DO 450 J=1,JM. The statement

IF(IEXT.EQ.ISPLIT) GO TO 8000

ensures that the last time step is not included in the averaging. This is because it is included as part of the initialization at the start of the external mode time step loop, in the loops DO 401 and DO 400.

The external mode time step is ended with the statement 8000 CONTINUE, and the internal mode calculations are resumed. The command

IF(MODE.EQ.2) GO TO 8200

ensures that if the model is being run in the barotropic mode, the three dimensional calculations are not performed. The command

IF(IINT.EQ.1.AND.TIME0.EQ.0.) GO TO 8200

ensures that if it is the first internal mode time step (IINT=1), the model jumps to statement 8200 CONTINUE and does not perform the three dimensional calculations, since the values of these initial variables were given in the initialization before the numerical integration.

The internal and external modes have different truncation errors, so that the vertical integrals of the internal mode velocities are not the exact values calculated for the barotropic velocities. The model now adjusts U(I,J,K) and V(I,J,K) such that their vertical averages are equal to UA(I,J) and VA(I,J). The temporary storage array TPS is set to zero to initialize the vertical integration of the velocity U(I,J,K), which is performed by the command

TPS(I,J)=TPS(I,J)+U(I,J,K)*DZ(K).

The loop DO 302 K=1,KBM1 that performs the calculation

U(I,J,K)= (U(I,J,K)-TPS(I,J)) + (UTB(I,J)+UTF(I,J))/(DT(I,J)+DT(I-1,J))

subtracts the vertically averaged velocity stored in TPS from U(I,J,K) to obtain the deviation from the mean. It then adds the average of UTB(I,J) and UTF(I,J) divided by the depth (recall that when UTF is calculated in the loop DO 450, it is defined as a velocity multiplied by the depth). This procedure ensures that the vertical average of U(I,J,K) is equal to UA(I,J). The three DO loops through statement numbers 303, 304, and 306 perform the analogous calculation for the V(I,J,K) velocity.

The vertical velocity is determined with the command CALL VERTVL(DTI2), and the command CALL BCOND(5) applies the boundary conditions (masking) to the vertical velocity.

The model now calculates twice the turbulent kinetic energy Q2(I,J,K) and the turbulence macroscale Q2L(I,J,K) at the forward time step. If the model followed the usual naming convention for the leapfrog time step, these arrays would be named Q2F and Q2LF. However, to save array space, these forward time step variables are stored temporarily in arrays UF and VF, respectively. First these arrays UF and VF are initialized to zero.

The statement CALL ADVQ(Q2B,Q2,DTI2,UF) has input parameters Q2B and Q2, twice the turbulent kinetic energy at the back and centered time steps, and twice the internal mode time step DTI2. The returned parameter is the advection for twice the turbulent kinetic energy at the forward time step, stored in array UF.

The statement CALL ADVQ(Q2LB,Q2L,DTI2,VF) has input parameters Q2LB and Q2L, for the back and centered time steps, and the time step DTI2. The returned parameter is the advection for variable at the forward time step, stored in array VF.

The command CALL PROFQ(DTI2) calls this subroutine to solve for the vertical profile of the turbulent kinetic energy and turbulence macroscale, still stored as UF and VF.

The command CALL BCOND(6) applies the boundary conditions (masking) to the turbulent kinetic energy and turbulence macroscale, still stored as UF and VF. Most of the model variables that are integrated in time are filtered with an Asselin (1972) filter to prevent the solutions at odd and even time steps from diverging. The turbulent Kinetic energy and turbulent length scale are filtered over the three time steps (Q2B,Q2,UF) and (Q2LB,Q2L,VF) using the commands

```
Q2=Q2+0.5*SMOTH*(UF+Q2F-2.0*Q2)
Q2L=Q2L+0.5*SMOTH*(VF+Q2LB-2.0*Q2L),
```

and the time sequence is reset with the commands

```
Q2B=Q2
Q2=UF
Q2LB=Q2L
Q2L=VF.
```

The model next computes the temperature and salinity. If the model followed the usual naming convention for the leapfrog time step, the variables would be named TF and SF. However, these variables will be stored in the arrays UF and VF to save space. The command

```
IF(MODE.EQ.4) GO TO 360
```

is used to skip over these calculations for temperature and salinity when the model is being run in the three dimensional diagnostic mode, which holds the temperature and salinity (and hence density) to their initial values. The command

```
CALL ADVT(TB,T,TMEAN,TSURF,DTI2,UF)
```

calls the horizontal scalar advection subroutine, with the input parameters of the temperature at the back (TB) and centered (T) time steps, the mean temperature (TMEAN), the surface temperature (TSURF), and twice the internal mode time step (DTI2). The subroutine uses the leapfrog time step to calculate the temperature at the forward time step, and stores it in array UF as the returned parameter. The command

```
CALL ADVT(SB,S,SMEAN,SSURF,DTI2,VF)
```

calls the advection subroutine with the input parameters of the salinity at the back (SB) and centered (S) time steps, the mean salinity (SMEAN), the surface salinity (SSURF), and twice the internal mode time step. The returned parameter is the salinity at the forward time step, stored in array VF. The commands

```
CALL PROFT(UF,WTSURF,DTI2)
CALL PROFT(VF,WSSURF,DTI2)
```

call this subroutine to determine the vertical profiles of scalar quantities. The input parameters are the surface temperature and salinity fluxes, WTSURF and WSSURF, the time step, and the temperatures and salinities stored in UF and VF. The vertical profiles of temperature and salinity are calculated and returned in UF and VF. The command CALL BCOND(4) calls the subroutine to apply the boundary conditions and land mask to the temperature and salinity, stored as UF and VF. The temperature and salinity are filtered over the three time steps using an Asselin (1972) filter with the commands

```
T=T+0.5*SMOTH*(UF+TB-2.0*T)
S=S+0.5*SMOTH*(VF+SB-2.0*S),
```

and the time sequence is reset with the commands

```
TB=T
T=UF
SB=S
S=VF.
```

The density is calculated with the statement CALL DENS. This marks the end of the

prognostic calculations for temperature, salinity, and density. These calculations would have been skipped in the diagnostic mode (MODE=4).

The last set of model prognostic calculations is for the velocities UF and VF. The commands

```
CALL ADVU(DRHOX,ADVUU,DTI2)
CALL ADVV(DRHOY,ADVVV,DTI2)
```

call the subroutines to calculate the horizontal changes for the U and V velocities. The calling parameters are the baroclinic pressure gradient terms (DRHOX, DRHOY), the vertical integral of the horizontal dispersion terms (ADVUU, ADVVV), and twice the internal mode time step DTI2. These subroutines include the influences of horizontal advection and diffusion, Coriolis, and baroclinic pressure gradient terms. The centered difference scheme is used to calculate UF and VF. The commands CALL PROFU(DTI2) and CALL PROFV(DTI2) call the subroutines to calculate the vertical profile of the internal mode velocities UF and VF. The command CALL BCOND(3) is used to apply the boundary conditions and velocity masks to UF and VF. These velocities are then filtered over the time steps with an Asselin (1972) filter.

The time series is reset with the commands

```
UB(I,J,K)=U(I,J,K)
U(I,J,K)=UF(I,J,K)
VB(I,J,K)=V(I,J,K)
V(I,J,K)=VF(I,J,K)

EGB(I,J)=EGF(I,J)
ETB(I,J)=ET(I,J)
ET(I,J)=ETF(I,J)
DT(I,J)=H(I,J)+ET(I,J)
UTB(I,J)=UTF(I,J)
VTB(I,J)=VTF(I,J).
```

This concludes all the time step calculations for the numerical integration of the internal mode time step. The command CALL FINDPSI is used to calculate the stream function PSI(I,J). The remaining commands are to print out or save information to various files at selected intervals, and the user will modify these to suit the application. For example, the command

```
IF(MOD(INT,IPRINT).NE.0) GO TO 7000
```

is used to print out various fields every IPRINT number of internal mode time steps IINT. The command

```
IF(ABS(VAMAX).GT.100.0.OR.ABS(VAMIN).GT.100.0) STOP
```

is used to stop the caclulation in case of numerical instabilities. The statement 9000 CONTINUE ends the internal mode time step DO loop.

For long time simulations of the model, it may be desirable to write model data to a restart file. It is best to write binary data files for greater precision. Since the model uses a centered difference (leapfrog) time step, two time levels are necessary for a 'seemless' restart. This can be done with the command

```
      WRITE(77) TIME,
     1  CURV42D,WUBOT,WVBOT,AAM2D,UA,UAB,VA,VAB,EL,ELB,
     2  ET,ETB,EGB,UTB,VTB,U,UB,W,V,VB,T,TB,S,SB,RHO,ADVUU,
     3  ADVVV,ADVUA,ADVVA,KM,KH,KQ,L,Q2,Q2B,AAM,Q2L,Q2LB.
```

5. THE COMPUTATIONAL MOLECULE

The model equations are finite differenced on an Arakawa-C staggered grid, shown in Fig. 1 for the external mode calculation. The ocean depths H(I,J) and sea level EL(I,J) are specified or calculated at the center point of a grid square. The east-west velocity UA(I,J) is calculated on the western side of the grid square, and the north- south velocity VA(I,J) is calculated on the south side of the grid square. For three dimensional calculations, U(I,J,K) is calculated on the west side, V(I,J,K) on the south side, and T(I,J,K), S(I,J,K), and RHO(I,J,K) at the center of the grid square at each intermediate level ZZ(K). The vertical velocity W(I,J,K), turbulent kinetic energy Q2(I,J,K), turbulent length scale L(I,J,K), their product Q2L(I,J,K), and diffusivities KM(I,J,K) and KH(I,J,K) are all calculated at the center point of a grid square, but at the Z(K) level in the vertical.

When a variable is calculated at a grid point from the finite difference equations, some variables at surrounding grid points are used in the calculation. This pattern of influence of the surrounding grids in the calculation of the particular variable is called the "computational molecule", and depends on the form of the finite difference equations. The computational molecule for the sea level, temperature, salinity, and other variables calculated at the center of a grid square is shown in Fig. 2. The computational molecules for the U and V velocities are shown in Figs. 3 and 4, respectively. These latter two molecules reach out farther to adjacent grid points because of the form of the momentum advection and diffusion terms.

The calculations are carried out over the entire model domain, over both land and ocean points. This is why land points are set to a small nonzero value (typically less than one meter), so that division by zero does not occur when dividing by D(I,J)=H(I,J)+EL(I,J) in the calculations. When calculations are performed at the boundary gridpoints, or interior points close to the boundary, the computational molecule reaches out to values outside the domain, and the bounds of some arrays may be exceeded. The user should check the compiler options for the particular computer being used, to see that this is allowed. When this situation occurs, the computer selects some value in an adjacent array stored in the COMMON block memory location. This is why all variables in the COMMON block are initialized to zero or some value at the start of the program and redefined later: to avoid an "undefined variable" error message if a calculation exceeds the bounds of an array. When a calculation makes use of values from outside the bounds of an array, an erroneous value is produced. The program must then replace this value before subsequent calculations are made, to avoid this error propagating inward from the boundaries. If land points are at the edge of the domain this is done by multiplying by the appropriate land or velocity mask FSM(I,J), DUM(I,J), or DVM(I,J). If open boundaries are at the edge of the domain, the open boundary conditions must overwrite any spurious values.

The model equations are linearized near the boundaries to prevent the computational molecule there from extending outside the domain. Also, for many applications, it is desirable to apply linear open boundary conditions. At the eastern and western boundaries, this is done by setting

```
ADVUA(1,J)=0
ADVUA(IM,J)=0
ADVVA(1,J)=0
ADVVA(IM,J)=0
```

in subroutine ADVAVE. At the northern and southern boundaries the computational molecule does not extend outside the domain because the variables are calculated at J=2,...,JMM1, or J=3,...,JMM1 (where JMM1 = JM–1) in the MAIN program and subroutines. The values at the boundary are then specified explicitly as open boundary conditions at J=1 and J=JM (and sometimes at J=2 and J=JMM1 depending on the application). Fig. 1 is a schematic of the Arakawa-C grid and indicates which variables must be calculated linearly at the boundary, and which variables must be specified explicitly as open boundary conditions.

The model is set up with the assumption that the western boundary is a wall, and should run without difficulty in its present form when appropriate boundary conditions are specified. Should the user change the model equations, such as by adding a new diffusion algorithm that changes the computaional molecule, or by configuring the model to a new domain, it is his responsibility to ensure that any spurious values resulting from computations near the boundaries are overwritten by the masks or appropriate boundary conditions.

6. SUBROUTINE BCOND

The boundary conditions are applied in subroutine BCOND. Basically, this subroutine is divided into six independent sections that apply the appropriate boundary conditions to the variables. This subroutine is called from the MAIN program with the command

CALL BCOND(IDX)

where the calling parameter IDX is an integer from 1 to 6. Then at the start of subroutine BCOND the computed GO TO branch statement

GO TO (10,20,30,40,50,60), IDX

uses this value of IDX to direct the program to the appropriate section where the boundary conditions are applied. After the specification of any open boundary conditions, the variables at land regions are set to zero by multiplying by the appropriate land FSM(I,J) or velocity DUM(I,J), DVM(I,J) mask, and control is returned to the MAIN program with a RETURN statement. The masking is done after the open boundary conditions are applied, because they will then overwrite any specifications made at a land boundrary. In general, it is not necessary to specify boundary conditions at a land boundary, since the masking will automatically set to zero the normal component of velocity there.

For some problems such as a hurricane passage through the Gulf of Mexico, the wind stress and atmospheric pressure must be specified as a function of time. This can be done conveniently by adding a section to subroutine BCOND for the atmospheric forcing. At present the subroutine BCOND has six sections for the boundary conditions on the model variables.

The Sea Level

For IDX=1, boundary conditions are applied to the external mode sea level ELF(I,J). For some applications, such as when the open boundary is along a shelf break, the sea level can be set to zero. In many cases the sea level is set equal to the upstream value, especially at outflow boundaries. It is often useful to apply radiation conditions (Chapman 1985) at cross shelf boundaries. If it is desired to explicitly give values at a north, east, or south boundary, then boundary arrays ELN(IM), ELE(JM), or ELS(IM) can be specified, and the value of ELF(I,J) set equal to these values at the boundary. The sea levels are masked with the land mask ELF(I,J)=ELF(I,J)*FSM(I,J).

The External Mode Velocities

For IDX=2, boundary conditions are applied to the external mode velocities UAF(I,J) and VAF(I,J). For some applications such as modeling the Gulf Stream or Gulf of Mexico, the inflow (and sometimes the outflow) velocities can be specified. This can be done by specifying the boundary arrays UABE(JM), VABN(IM), and VABS(IM). At outflow boundaries, the normal component of velocity can be set to the value immediately upstream. For some applications, radiation conditions can be used. The velocities are masked by the arrays

UAF(I,J)=UAF(I,J)*DUM(I,J)

VAF(I,J)=VAF(I,J)*DVM(I,J).

At a land boundary, the masking by DUM(I,J) and DVM(I,J) automatically applies the condition that the normal component of velocity vanishes there.

The Internal Mode Velocities

For IDX=3, boundary conditions are applied to the internal mode velocities UF(I,J,K) and VF(I,J,K). Usually they are radiation conditions or upstream conditions. For some applications, the inflow velocity could be specified as a function of depth. The velocities are masked with the arrays

UF(I,J,K)=UF(I,J,K)*DUM(I,J)

VF(I,J,K)=VF(I,J,K)*DVM(I,J).

The masking automatically applies the condition that the normal component of velocity vanishes at the boundary.

The Temperature and Salinity

For IDX=4, boundary conditions are applied to the temperature TF(I,J,K) and salinity SF(I,J,K), which are being stored in the arrays UF(I,J,K) and VF(I,J,K), respectively.

The temperature can be specified on the boundaries with the boundary value arrays TBN(IM,KB), TBE(JM,KB), and TBS(IM,KB). These arrays can be set to the appropriate values of climatology, with 10 C^o subtracted. At the boundary the value of TF(I,J,K), stored in UF(I,J,K), is set equal to the boundary value array.

The salinity can be specified on the boundaries with the boundary value arrays SBN(IM,KB), SBE(JM,KB), and SBS(IM,KB). These arrays can be set to the appropriate values of climatology with 35 ppt subtracted. At the boundary the value of SF(I,J,K), stored in VF(I,J,K), is set equal to the boundary value arrays. The temperatures and salinities are often held to a constant value at an inflow boundary, and set to upstream values and an outflow boundary. Both temperature and salinity are masked with the land mask FSM.

The Vertical Velocity

For IDX=5, boundary conditions are applied to the sigma coordinate vertical velocity W(I,J,K). The vertical velocity is masked with the land mask FSM(I,J). For some applications, the vertical velocity on the eastern boundary W(IM,J,K) is set to zero.

The Turbulent Kinetic Energy

For IDX=6, boundary conditions are applied to the turbulent kinetic energy Q2F(I,J,K) stored in UF(I,J,K), and twice the turbulent kinetic energy times the turbulence length scale Q2LF(I,J,K) stored as VF(I,J,K). They are masked with the land mask FSM(I,J). For some applications, these values are set equal to a constant at an inflow boundary.

7. UTILITIES SUBROUTINES

The model has several subroutines for printing out data fields for ready visualization. Their use is described briefly at the beginning of each print subroutine. Here we describe their use in more detail. It should be kept in mind that some earlier versions of the model may have different, but similar subroutines for printing output. Here we describe the subroutines used in the Princeton model pmod.f.

SUBROUTINE PRXY

This subroutine prints two dimensional (X,Y) arrays, and is probably the most frequently used print routine. The calling parameters are given

SUBROUTINE PRXY(LABEL,TIME,ARRAY,IM,ISKIP,JM,JSKIP,SCALA)

LABEL	The name of the field to appear on the printout, passed in quotes as a character string.
TIME	The variable TIME, giving elapsed time in days. At initialization, be sure to set TIME=0.0 to use these subroutines.
ARRAY	The variable to be printed, of dimensions (IM,JM), eg., H for the ocean depths, EL for the surface elevation.
IM	The first dimension of the array in the X–direction.
ISKIP	The frequency at which variables in the X–direction are skipped when printing. To print every third value, set ISKIP=3; to print every other value, set ISKIP=2; to print every value, set ISKIP=1.
JM	The second dimension of the array in the Y–direction.
JSKIP	The frequency at which variables in the Y–direction are skipped when printing. To print every third value, set JSKIP=3; to print every other value, set JSKIP=2, to print every value, set JSKIP=1.
SCALA	The scale used to print values. The subroutine prints out the actual values divided by the scale. The subroutines also print out the message "MULTIPLY BY" and the value of the scale. If SCALA is zero, the scale is computed by the subroutine, which is useful if the magnitude of the variable is unknown.

The use of the scale is illustrated here with several useful examples. To display the grid array on the order of several tens of thousands of meters, say DX(I,J)=10000.0, set SCALA=1.E+3, and the PRXY subroutine will print out an array of numbers, each of value 10, corresponding to kilometers.

To display currents which are typically less than one meter per second, say UAB(I,J)=0.30, set SCALA=1.E–2, and the PRXY subroutine will print out an array of numbers each of value 30, corresponding to centimeters per second.

To display sea levels that are typically less than one meter, say ELB(I,J)=0.05, set SCALA=1.E–2, and PRXY prints out an array of numbers each with value 5, corresponding to sea level in centimeters.

To display the wind stress (divided by the water density) in units of $m^2 s^{-2}$, say WUSURF(I,J) = 2.0E–4, set SCALA=1.E–4, and PRXY prints out an array of numbers each of value 2, corresponding to wind stress in $dynes\, cm^{-2}$ (divided by the water density $1.0\, gm\, cm^{-3}$).

The SCALA is set similarly in the other print subroutines below.

SUBROUTINE PRXYZ

This subroutine prints horizontal (X–Y) slices of a three dimensional array. The calling parameters, besides those discussed above, are

SUBROUTINE PRXYZ(LABEL,TIME,ARRAY,IM,ISKIP,JM,JSKIP,KB,SCALA).

ARRAY	A three dimensional array, of dimensions (IM,JM,KB), eg., T for temperatures, S for salinities.
KB	The dimension in the vertical.

The subroutine internally choses the X–Y slices to print. This is done with the statements of the form

```
DIMENSION KP(3)
DATA KE,KP/3,1,3,5/
```

which in this case dimensions an array KP(KE) and specifies that the levels to be printed will be K=1,3,5. To print out five levels, K=1,5,7,9,11, the user could modify the subroutine with the statements

```
DIMENSION KP(5)
DATA KE,KP/5,1,5,7,9,11/.
```

SUBROUTINE PRXZ

This subroutine prints vertical (X–Z) slices of a three dimensional array, for the slices at J=J1, J2. The output is also displayed with the number K of the layer and the value ZZ(K) printed in columns to the left. The value of the depth is printed as a row across the top. The calling parameters, besides those discussed above are

SUBROUTINE PRXZ(LABEL,TIME,ARRAY,IM,ISKIP,JM,J1,J2,KB,SCALA,DT,ZZ).

J1,J2	The Y values (at J=J1,J2) at which the the slice in the X–Z plane are taken.
DT	The depth of the water column, array DT(IM,JM).
ZZ	The array specifying the midpoints of the levels ZZ(KB).

SUBROUTINE PRYZ

This subroutine prints vertical (Y–Z) slices of a three dimensional array. This version of the subroutine prints the slices at I=IM/2 and I=IM–3. In this subroutine ISKIP is used only in the determination of the scale if SCALA=0. The subroutine also prints out the value K of the vertical layer and corresponding value ZZ(K) to the left of the output, and the depth in meters in a row across the top. The calling parameters are

SUBROUTINE PRYZ(LABEL,TIME,ARRAY,IM,ISKIP,JM,JSKIP,KB,SCALA,DT,ZZ).

As a final note, these subroutines internally make use of the one dimensional arrays IDT, JDT, LINE, and NOM to set up the printing layout on the pages. These arrays must be dimensioned to the same value which is greater than both IM and JM, or an error message will result. It is usually best to dimension these arrays to some high value, say IDT(500), JDT(500), LINE(500), and NOM(500) to prevent the necessity of frequent changes.

8. BOX MODEL TEST CASE

This is a box model test case to demonstrate the application of the Princeton model. It is a rectangular domain of dimension 2000 km in the east - west direction, 1000 km in the north - south direction, and of constant depth 2000 m. A simple sinusoidal wind stress profile is set up over the basin with easterly winds over the southern part and westerly winds over the northern part, that remain constant in time. The model has $20 \times 20 \times 24 = 9600$ total node points so that it can be run on most computers. The specifications are listed below.

(1) The box model dimensions are

IM = 20, JM = 20, KB = 24.

(2) A rectangular cartesian domain is used with grid spacing

DX(I,J) = 100.E3 (100 Km east - west)

DY(I,J) = 50.E3 (50 Km north - south).

(3) The coriolis parameter varies linearly with a beta plane approximation

BETA = 1.98E-11

COR4(I,J)=0.25*(7.292E-5+BETA*(J-1.0)*DY(I,J)).

(4) A constant depth of 2000 m is specified so that H(I,J) = 2000. To indicate a land boundary (the walls of the box model) we set

H(I,1) = H(I,JM) = 0.5 for I=1,...,IM

H(1,J) = H(IM,J) = 0.5 for J=1,...,JM.

(5) The temperature is specified with the exponentialy decreasing profile, with the first six levels at a constant value

TB(I,J,K)=10.0*EXP(ZZ(K))+10.0

IF(K.LT.6) TB(I,J,K)=TB(I,J,6).

The salinity is specified to be a constant everywhere SB(I,J,K)=35. It is necessary to subtract 10 degrees Celsius from the temperature and 35 parts per thousand from the salinity as part of the initialization. These values are added back when computing the density in SUBROUTINE DENS.

TB(I,J,K)=TB(I,J,K)-10.

```
      SB(I,J,K)=SB(I,J,K)-35.
```

A call is made to subroutine DENS to determine the density RHO(I,J,K), and for this example, the value of the mean density array is set equal to RMEAN(I,J,K) = RHO(10,5,K). The mean values of temperature and salinity are set equal to TMEAN(I,J,K) = TB(I,J,K), and SMEAN(I,J,K) = SB(I,J,K).

(6) For an enclosed domain, it is not necessary to specify any boundary value arrays. The requirement that the normal component of velocity must vanish on a boundary is applied with the DUM(I,J) and DVM(I,J) velocity masks in SUBROUTINE BCOND, and this will overwrite any specifications using the boundary value arrays.

(7) The momentum fluxes WUSURF(I,J) and WVSURF(I,J) are the negative of the wind stress, divided by the density of water, expressed in the MKS units of $m^2 s^{-2}$. They are typically of magnitude 1.E–4. This example uses a simple sinusoidal profile with momentum input from easterly winds over the southern part of the domain, and from westerly winds over the northern part.

```
      WUSURF(I,J)=(1.E-4*COS(PI*(J-1)/JMM1))
     1*0.25*(DVM(I,J+1)+DVM(I-1,J+1)+DVM(I-1,J)+DVM(I,J))
      WVSURF(I,J)=0.
```

The average over the velocity mask reduces the value of the wind stress at the boundary, to reduce boundary effects.

(8) The surface temperature and salinity fluxes will be set to zero with the commands WTSURF(I,J)=0, and WSSURF(I,J)=0.

(9) The model run will use an external model time step (DTE) of 192 s, and an internal mode time step (DTI) of 5760 s. This is set in the model by the commands DTI=5760 and ISPLIT=30.

(10) The initial time is set to zero with the command TIME=0. The model will be set to run 5 days with the command IDAYS=5, and the print intervals will be specified with this same value, so that there will be only one printout to the standard out (UNIT=6) at the end of the model run IPRTD1=5 and IPRTD2=5. In this application the value of ISWITCH is not used. The data can be written to files to be saved as desired (see User Supplied Input).

(11) The three dimensional calculation will be made by setting the parameter MODE=3.

(12) The vertical kinematic viscosity KM(I,J,K), vertical heat diffusivity KH(I,J,K), turbulent kinetic energy Q2B(I,J,K), and turbulence macroscale Q2LB(I,J,K) are initialized to zero or some small value

```
      Q2B(I,J,K)=0

      Q2LB(I,J,K)=0
```

KM(I,J,K)=2.E–4

KM(I,J,1)=KM(I,J,KB)=0

KH(I,J,K)=KM(I,J,K)

and the horizontal kinematic viscosity is initialized to the value AAM(I,J,K)=50.0.

The model is initialized from a state of rest so that the initial velocities and sea level are zero. This is taken care of in the initialization at the start of the MAIN program, where all values are set to zero before the values for each application are specified or read in from an input file.

(13) The PERIOD of time in days over which the calculations are spun up, or RAMP–ed, is one inertial period, that value of the inertial period at the center of the domain

PERIOD = DAYI*(2.0*PI)/(COR4(IM/2,JM/2)*4.0).

The other parameters in the model can usually be kept at the values originally specified, e.g.,

ISPADV=1

TPRNU=1.0

SMOTH=0.10

HORCON=0.02.

(14) Here we display the model parameters and model output, making use of the utilities print subroutines. First we wish to see the vertical levels and spacing. This is determined in subroutine DEPTH and printed from that subroutine. The calling parameters are KB and KBM1 and are given in the call statement

CALL DEPTH(Z,ZZ,DZ,DZZ,DZR,KB,KBM1).

Some earlier versions of subroutine DEPTH may have a different parameter list, but the Z, ZZ, DZ, and DZZ are always determined. The model output that is printed to the standard out (UNIT=6) is shown in Fig. 5.

The constant model depths of 2000 m in array H(I,J) can be displayed with the call

CALL PRXY('TOPOGRAPHY',TIME,H,IM,2,JM,1,1.E0) .

The array will be printed with every other value of I omitted, and the results are shown in Fig. 6. Since the scale is 1.E0 the values 2000 are printed. The land values along the boundary were set to the value 0.5 m, and so are printed as 0 when this scale is used. In

general, the variables over land points are set to zero when multiplied by the land mask in subroutine BCOND.

The CFL stability criterion time step for the model was calculated and stored in the temporary storage space array TPS(I,J) and can be displayed with the call

CALL PRXY('CFL TIME STEP',TIME,TPS,IM,2,JM,1,1.E0).

The values in the array are the constant 159 s, since the depths and grid spacings were held constant (Fig. 7). However, the CFL calculation is useful when setting up a model for any given problem.

The Coriolis parameter which varies with the latitude of each grid box can be displayed with the call statement

CALL PRXY('COR',TIME,COR,IM,2,JM,1,1.E-8).

A typical value of 8082 printed out in Fig. 8 means that the value of the Coriolis parameter is $8082 \times 10^{-8}\ s^{-1}$ since the scale is 1.E-8. Some versions of the model use the variable COR4(IM,JM) which is the value of the Coriolis parameter divided by 4.

The vertical distribution of initial temperature can be displayed with the call statement

CALL PRXZ('TEMP.',TIME,TB,IM,2,JM,5,10,KB,1.E-1,DT,ZZ).

which prints out two cross sections in the X-Z plane at the rows J=5, 10. Since these are identical because of the uniform initial conditions, only one slice is shown in Fig. 9. Since the SCALE=1.E-1, a typical value of 88.2 printed out means that the model temperature is $88.2 \times 10^{-1}\ C^{o}$ or $8.82\ C^{o}$. Since 10^{o} was subtracted from the temperature as part of the initialization, this represents a true temperature of $18.82\ C^{o}$. Note that the bottom layer K=KB always has zero values.

The initial density distribution can be displayed with the call statement

CALL PRYZ('RHO',TIME,RHO,IM,2,JM,2,KB,1.E-4,DT,ZZ).

This subroutine prints out two cross sections in the Y-Z plane, at I=IM/2=10 and I=IM-3=17. (The value of ISKP=2 is not used when the scale is specified as nonzero.) Because the cross sections are identical only one is shown here in Fig. 10. Since the scale is 1.E-4, a value in the printout of 1.4 means that the model value in array RHO is 1.4×10^{-4}. Since the model uses a normalized density for greater precision when calculating baroclinic pressure gradients, the actual physical density is found by adding 1.025 and multiplying the result by 1000, or

$$(1.025 + 0.00014) \times 1000 = 1025.14\ Kg\ m^{-3}.$$

Note that the bottom layer K=KB always has zero values.

The model output after the five day run can be examined. The surface momentum flux in the east-west direction driving the model can be displayed with the call statement

CALL PRXY('WUSURF',TIME,WUSURF,IM,2,JM,1,1.0E-8)

and the results are shown in Fig. 11. Since the surface momentum flux input WUSURF has

the opposite sign as the wind stress, the momentum input from the easterly winds over the southern part of the domain is positive, while the momentum input from the westerly winds over the northern part is negative. Since the scale is 1.0E–8, a typical value of 8794 means that the wind stress divided by the water density is $8794 \times 10^{-8}\ m^2 s^{-2}$ resulting from a wind stress of magnitude 0.8794 dynes cm^{-2}.

The sea level of the gyre that is set up by the forcing can be displayed by the call statement

```
CALL PRXY('FREE SURFACE',TIME,ELB,IM,2,JM,1,1.E–4).
```

The print out is shown in Fig. 12. Since the scale is 1.E–4, a value of 230 means that the sea level is $230 \times 10^{-4}\ m$, or 2.3 cm.

The current structure of this gyre can be displayed by the calls

```
CALL PRXY('AVERAGE U COMP.',TIME,UAB,IM,2,JM,1,1.E–4)
CALL PRXY('AVERAGE V COMP.',TIME,VAB,IM,2,JM,1,1.E–4)
```

and this is shown in Figs. 13 and 14. The scale is 1.E–4 so that a printed value of 124 means that the current velocity is $124 \times 10^{-4}\ ms^{-1}$, or 1.24 $cm\ s^{-1}$.

REFERENCES

Asselin, R. (1972) Frequency Filters for Time Integrations, *Mon. Weather Rev.*, **100**, 487–490.

Blumberg, A.F. (1977) Numerical Tidal Model of Chesapeake Bay, *Journal of the Hydraulics Division, ASCE*, **103**, 1–10.

Blumberg, A.F., and H.J. Herring (1987) Circulation Modelling Using Orthogonal Curvilinear Coordinates, In: Nihoul, J.C.J. and Jamart, B.M. (editors), *Three–Dimensional Models of Marine and Estaurine Dynamics*, Elsevier, New York, pp. 55–88.

Blumberg, A.F., and L.H. Kantha (1985) Open Boundary Condition for Circulation Models, *Journal of Hydraulic Engineering, ASCE*, **111**, 237–255.

Blumberg, A.F., and G.L. Mellor (1983) Diagnostic and Prognostic Numerical Circulation Studies of the South Atlantic Bight, *Journal of Geophysical Research*, **88** (C8), 4579–4592.

Blumberg, A.F., and G.L. Mellor (1985) A Simulation of the Circulation in the Gulf of Mexico, *Israel Journal of Earth Sciences*, **34**, 122–144.

Blumberg, A. F., and G.L. Mellor (1987) A Description of a Three–Dimensional Coastal Ocean Circulation Model, In: N.S. Heaps (Editor), *Three Dimensional Coastal Ocean Models, Coastal and Estuarine Sciences*, **4**, American Geophysical Union, Washington, D.C., pp. 1-16.

Blumberg, A.F., and L.–Y. Oey (1985) Modeling Circulation and Mixing in Estuaries and Coastal Oceans, In: S. Manabe (Editor), Issues in Atmospheric and Oceanic Modeling, Part A, Climate Dynamics, *Advances in Geophysics*, **28**, 525–547.

Chapman, D.C. (1985) Numerical Treatment of Cross–Shelf Open Boundaries in a Barotropic Coastal Ocean Model, *Journal of Physical Oceanography*, **15**, 1060–1075.

Galperin, B., and G.L. Mellor (1990a) A Time–Dependent, Three–Dimensional Model of the Delaware Bay and River System. Part 1: Description of the Model and Tidal Analysis, *Estaurine, Coastal and Shelf Science*, **31**, 231–253.

Galperin, B., and G.L. Mellor (1990b) A Time–Dependent, Three–Dimensional Model of the Delaware Bay and River System. Part 2: Three-Dimensional Flow Fields and Residual Circulation, *Estaurine, Coastal and Shelf Science*, **31**, 255–281.

Mellor, G.L. (1990) *Documentation for A Three-Dimensional, Primitive Equation, Numerical Ocean Model*, 41 pp., available from: Program in the Atmospheric and Oceanic Sciences, PO Box CN 710, Sayer Hall, Princeton University, Princeton, New Jersey, 08544–0710.

Mellor, G.L., and A.F. Blumberg (1985) Modeling Vertical and Horizontal Diffusivities with the Sigma Coordinate System, *Mon. Weather Rev.*, **113**, 1379–1383.

Mellor, G.L., and T. Ezer (1991) A Gulf Stream Model and an Altimetric Assimilation Scheme, *J. Geophys. Res.*, **96** (C5), 8779–8795.

Oey, L.–Y., G.L. Mellor, and R.I. Hires (1985a) A Three–Dimensional Simulation of the Hudson–Raritan Estuary. Part I: Description of the Model and Model Simulations, *J. Phys. Oceanog.*, **15**, 1676–1692.

Oey, L.–Y., G.L. Mellor, and R.I. Hires (1985b) A Three–Dimensional Simulation of the Hudson–Raritan Estuary. Part II: Comparisons with Observation, *J. Phys. Oceanog.*, **15**, 1693–1709.

Oey, L.–Y., G.L. Mellor, and R.I. Hires (1985c) A Three–Dimensional Simulation of the Hudson–Raritan Estuary. Part III: Salt Flux Analyses, *J. Phys. Oceanog.*, **15**, 1711–1720.

Shapiro, R. (1970) Smoothing, Filtering, and Boundary Effects, *Reviews of Geophysics and Space Physics*, **8** (2), 359–387.

LIST OF FIGURES

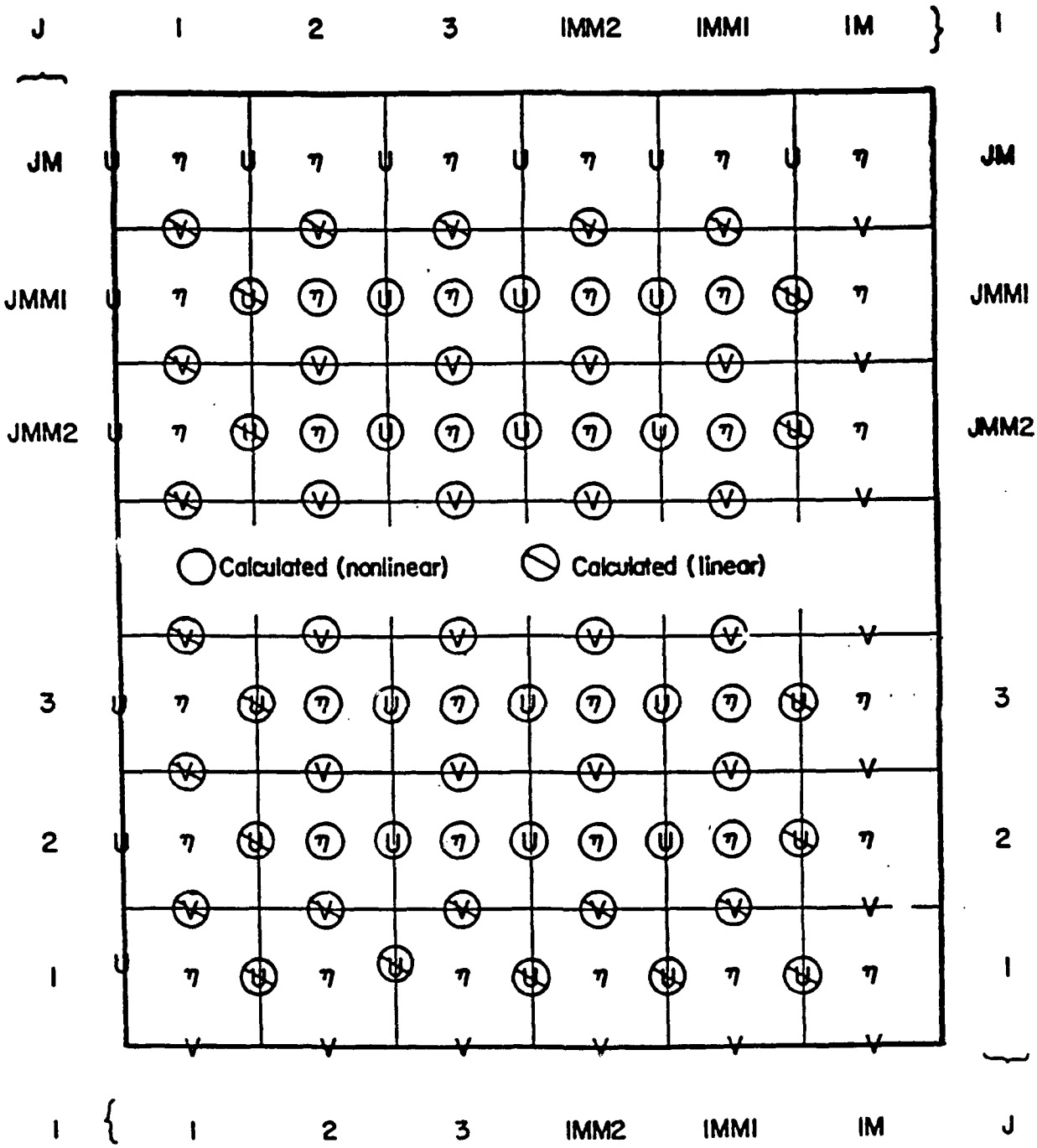

Fig. 1 The Arakawa-C staggard grid.

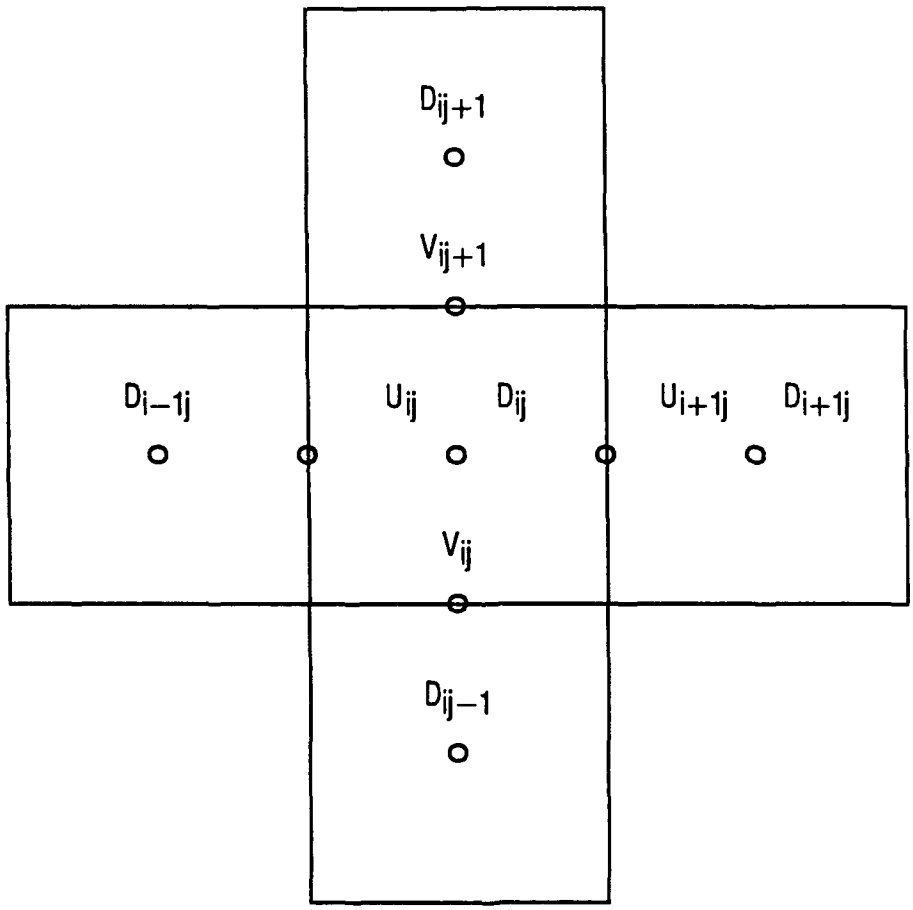

Fig. 2 The computational molecule for sea level, temperature, and other scalar variables.

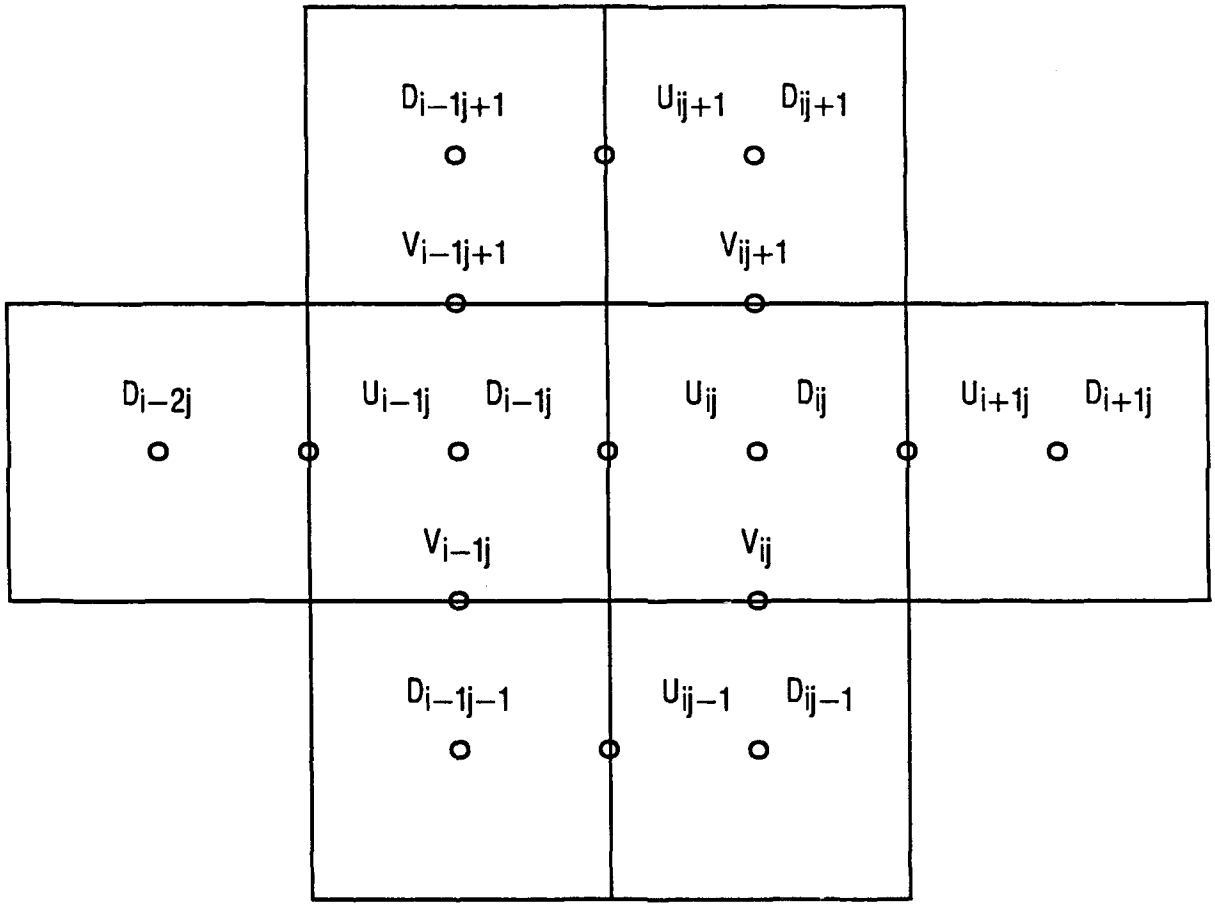

Fig. 3 The computational molecule for the U velocity.

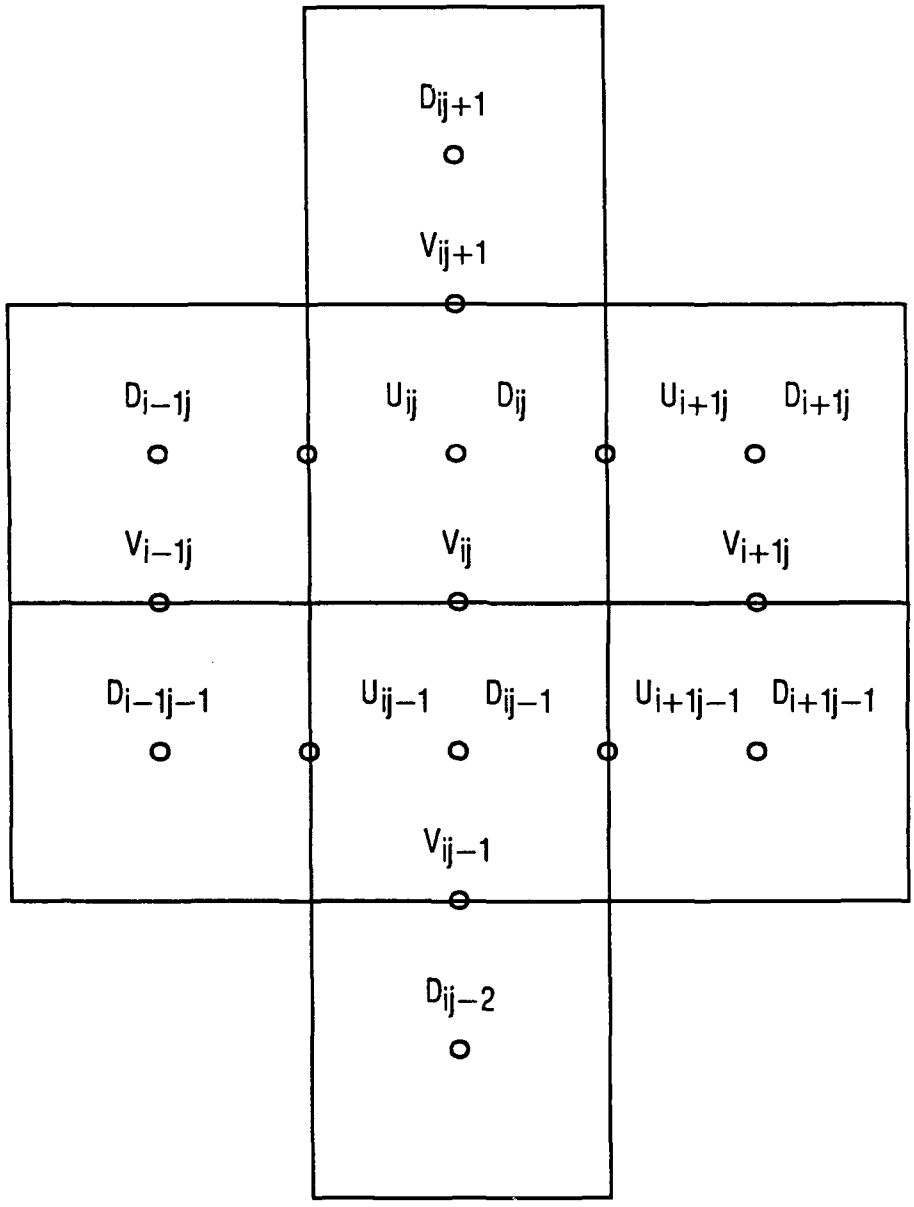

Fig. 4 The computational molecule for the V velocity.

K	Z	ZZ	DZ	DZZ
1	0.000	-0.003	0.006	0.006
2	-0.006	-0.009	0.006	0.009
3	-0.013	-0.018	0.013	0.018
4	-0.025	-0.035	0.025	0.035
5	-0.050	-0.071	0.050	0.054
6	-0.100	-0.125	0.050	0.050
7	-0.150	-0.175	0.050	0.050
8	-0.200	-0.225	0.050	0.050
9	-0.250	-0.275	0.050	0.050
10	-0.300	-0.325	0.050	0.050
11	-0.350	-0.375	0.050	0.050
12	-0.400	-0.425	0.050	0.050
13	-0.450	-0.475	0.050	0.050
14	-0.500	-0.525	0.050	0.050
15	-0.550	-0.575	0.050	0.050
16	-0.600	-0.625	0.050	0.050
17	-0.650	-0.675	0.050	0.050
18	-0.700	-0.725	0.050	0.050
19	-0.750	-0.775	0.050	0.050
20	-0.800	-0.825	0.050	0.050
21	-0.850	-0.875	0.050	0.054
22	-0.900	-0.929	0.050	0.046
23	-0.950	-0.975	0.050	0.050
24	-1.000	-1.025	0.000	0.000

Fig. 5 The box model vertical distribution of σ levels produced by a call to subroutine DEPTH.

TOPOGRAPHY

TIME = 0.0000 DAYS MULTIPLY ALL VALUES BY 1.000E+00

	1	3	5	7	9	11	13	15	17	19
20	1	1	1	1	1	1	1	1	1	1
19	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
18	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
17	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
16	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
15	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
14	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
13	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
12	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
11	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
10	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
9	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
8	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
7	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
6	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
5	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
4	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
3	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
2	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
1	1	1	1	1	1	1	1	1	1	1

Fig. 6 The box model 2000 m bathymetry, displayed by CALL PRXY('TOPOGRAPHY',TIME,H,IM,2,JM,1,1,E0).

CFL TIME STEP

TIME = 0.0000 DAYS MULTIPLY ALL VALUES BY 1.000E+00

	1	3	5	7	9	11	13	15	17	19
20	0	0	0	0	0	0	0	0	0	0
19	0	159	159	159	159	159	159	159	159	159
18	0	159	159	159	159	159	159	159	159	159
17	0	159	159	159	159	159	159	159	159	159
16	0	159	159	159	159	159	159	159	159	159
15	0	159	159	159	159	159	159	159	159	159
14	0	159	159	159	159	159	159	159	159	159
13	0	159	159	159	159	159	159	159	159	159
12	0	159	159	159	159	159	159	159	159	159
11	0	159	159	159	159	159	159	159	159	159
10	0	159	159	159	159	159	159	159	159	159
9	0	159	159	159	159	159	159	159	159	159
8	0	159	159	159	159	159	159	159	159	159
7	0	159	159	159	159	159	159	159	159	159
6	0	159	159	159	159	159	159	159	159	159
5	0	159	159	159	159	159	159	159	159	159
4	0	159	159	159	159	159	159	159	159	159
3	0	159	159	159	159	159	159	159	159	159
2	0	159	159	159	159	159	159	159	159	159
1	0	0	0	0	0	0	0	0	0	0

Fig. 7 The CFL criterion of 159 s for each grid square of the box model, displayed by CALL PRXY('CFL TIME STEP',TIME,TPS,IM,2,JM,1,1.E0).

COR

TIME = 0.0000 DAYS MULTIPLY ALL VALUES BY 1.000E-08

	1	3	5	7	9	11	13	15	17	19
20	9171	9171	9171	9171	9171	9171	9171	9171	9171	9171
19	9072	9072	9072	9072	9072	9072	9072	9072	9072	9072
18	8973	8973	8973	8973	8973	8973	8973	8973	8973	8973
17	8874	8874	8874	8874	8874	8874	8874	8874	8874	8874
16	8775	8775	8775	8775	8775	8775	8775	8775	8775	8775
15	8676	8676	8676	8676	8676	8676	8676	8676	8676	8676
14	8577	8577	8577	8577	8577	8577	8577	8577	8577	8577
13	8478	8478	8478	8478	8478	8478	8478	8478	8478	8478
12	8379	8379	8379	8379	8379	8379	8379	8379	8379	8379
11	8280	8280	8280	8280	8280	8280	8280	8280	8280	8280
10	8181	8181	8181	8181	8181	8181	8181	8181	8181	8181
9	8082	8082	8082	8082	8082	8082	8082	8082	8082	8082
8	7983	7983	7983	7983	7983	7983	7983	7983	7983	7983
7	7884	7884	7884	7884	7884	7884	7884	7884	7884	7884
6	7785	7785	7785	7785	7785	7785	7785	7785	7785	7785
5	7686	7686	7686	7686	7686	7686	7686	7686	7686	7686
4	7587	7587	7587	7587	7587	7587	7587	7587	7587	7587
3	7488	7488	7488	7488	7488	7488	7488	7488	7488	7488
2	7389	7389	7389	7389	7389	7389	7389	7389	7389	7389
1	7290	7290	7290	7290	7290	7290	7290	7290	7290	7290

Fig. 8 The Coriolis parameter in s^{-1} for each grid square of the box model, displayed by CALL PRXY('COR',TIME,COR,IM,2,JM,1,1.E-8).

TEMP.

TIME = 0.00 MULTIPLY ALL VALUES BY0.10E+00

SECTION J = 5

I =		1	3	5	7	9	11	13	15	17	19
	D =	1	2000	2000	2000	2000	2000	2000	2000	2000	2000
1	-0.003	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
2	-0.009	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
3	-0.018	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
4	-0.035	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
5	-0.071	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
6	-0.125	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2	88.2
7	-0.175	83.9	83.9	83.9	83.9	83.9	83.9	83.9	83.9	83.9	83.9
8	-0.225	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9	79.9
9	-0.275	76.0	76.0	76.0	76.0	76.0	76.0	76.0	76.0	76.0	76.0
10	-0.325	72.3	72.3	72.3	72.3	72.3	72.3	72.3	72.3	72.3	72.3
11	-0.375	68.7	68.7	68.7	68.7	68.7	68.7	68.7	68.7	68.7	68.7
12	-0.425	65.4	65.4	65.4	65.4	65.4	65.4	65.4	65.4	65.4	65.4
13	-0.475	62.2	62.2	62.2	62.2	62.2	62.2	62.2	62.2	62.2	62.2
14	-0.525	59.2	59.2	59.2	59.2	59.2	59.2	59.2	59.2	59.2	59.2
15	-0.575	56.3	56.3	56.3	56.3	56.3	56.3	56.3	56.3	56.3	56.3
16	-0.625	53.5	53.5	53.5	53.5	53.5	53.5	53.5	53.5	53.5	53.5
17	-0.675	50.9	50.9	50.9	50.9	50.9	50.9	50.9	50.9	50.9	50.9
18	-0.725	48.4	48.4	48.4	48.4	48.4	48.4	48.4	48.4	48.4	48.4
19	-0.775	46.1	46.1	46.1	46.1	46.1	46.1	46.1	46.1	46.1	46.1
20	-0.825	43.8	43.8	43.8	43.8	43.8	43.8	43.8	43.8	43.8	43.8
21	-0.875	41.7	41.7	41.7	41.7	41.7	41.7	41.7	41.7	41.7	41.7
22	-0.929	39.5	39.5	39.5	39.5	39.5	39.5	39.5	39.5	39.5	39.5
23	-0.975	37.7	37.7	37.7	37.7	37.7	37.7	37.7	37.7	37.7	37.7
24	-1.025	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Fig. 9 The vertical distribution of initial temperature for the slice at J=5 in the X–Z plane, displayed by CALL PRXZ('TEMP.',TIME,TB,IM,2,JM,5,10,KB,1,E-1,DT,ZZ). Add 10 C° to obtain the real temperature.

RHO

TIME = 0.00 MULTIPLY ALL VALUES BY0.10E-03

SECTION I = 10

J =		1	3	5	7	9	11	13	15	17	19
	D =	1.2000.	2000.	2000.	2000.	2000.	2000.	2000.	2000.	2000.	
1	-0.003	0.0	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
2	-0.009	0.0	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4	1.4
3	-0.018	0.0	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2	2.2
4	-0.035	0.0	3.7	3.7	3.7	3.7	3.7	3.7	3.7	3.7	3.7
5	-0.071	0.0	6.8	6.8	6.8	6.8	6.8	6.8	6.8	6.8	6.8
6	-0.125	0.0	11.5	11.5	11.5	11.5	11.5	11.5	11.5	11.5	11.5
7	-0.175	0.0	16.9	16.9	16.9	16.9	16.9	16.9	16.9	16.9	16.9
8	-0.225	0.0	22.2	22.2	22.2	22.2	22.2	22.2	22.2	22.2	22.2
9	-0.275	0.0	27.5	27.5	27.5	27.5	27.5	27.5	27.5	27.5	27.5
10	-0.325	0.0	32.8	32.8	32.8	32.8	32.8	32.8	32.8	32.8	32.8
11	-0.375	0.0	37.9	37.9	37.9	37.9	37.9	37.9	37.9	37.9	37.9
12	-0.425	0.0	43.1	43.1	43.1	43.1	43.1	43.1	43.1	43.1	43.1
13	-0.475	0.0	48.2	48.2	48.2	48.2	48.2	48.2	48.2	48.2	48.2
14	-0.525	0.0	53.2	53.2	53.2	53.2	53.2	53.2	53.2	53.2	53.2
15	-0.575	0.0	58.2	58.2	58.2	58.2	58.2	58.2	58.2	58.2	58.2
16	-0.625	0.0	63.1	63.1	63.1	63.1	63.1	63.1	63.1	63.1	63.1
17	-0.675	0.0	68.1	68.1	68.1	68.1	68.1	68.1	68.1	68.1	68.1
18	-0.725	0.0	72.9	72.9	72.9	72.9	72.9	72.9	72.9	72.9	72.9
19	-0.775	0.0	77.8	77.8	77.8	77.8	77.8	77.8	77.8	77.8	77.8
20	-0.825	0.0	82.6	82.6	82.6	82.6	82.6	82.6	82.6	82.6	82.6
21	-0.875	0.0	87.4	87.4	87.4	87.4	87.4	87.4	87.4	87.4	87.4
22	-0.929	0.0	92.5	92.5	92.5	92.5	92.5	92.5	92.5	92.5	92.5
23	-0.975	0.0	96.8	96.8	96.8	96.8	96.8	96.8	96.8	96.8	96.8
24	-1.025	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Fig. 10 The vertical distribution of initial density for the slice at I=10 in the Y–Z plane, displayed by
CALL PRYZ('RHO',TIME,RHO,IM,2,JM,2,KB,1.E-4,DT,ZZ).
The actual physical density in $Kg\ m^{-3}$ is found by adding 1.025 to the value and then multiplyng by 1000, eg., $(1.025 + 0.00014) \times 1000 = 1025.14\ Kg\ m^{-3}$.

WUSURF

TIME = 5.0000 DAYS MULTIPLY ALL VALUES BY 1.000E-08

	1	3	5	7	9	11	13	15	17	19
20	0	0	0	0	0	0	0	0	0	0
19	0	-4931	-4931	-4931	-4931	-4931	-4931	-4931	-4931	-4931
18	0	-9458	-9458	-9458	-9458	-9458	-9458	-9458	-9458	-9458
17	0	-8794	-8794	-8794	-8794	-8794	-8794	-8794	-8794	-8794
16	0	-7891	-7891	-7891	-7891	-7891	-7891	-7891	-7891	-7891
15	0	-6772	-6772	-6772	-6772	-6772	-6772	-6772	-6772	-6772
14	0	-5469	-5469	-5469	-5469	-5469	-5469	-5469	-5469	-5469
13	0	-4016	-4016	-4016	-4016	-4016	-4016	-4016	-4016	-4016
12	0	-2454	-2454	-2454	-2454	-2454	-2454	-2454	-2454	-2454
11	0	-825	-825	-825	-825	-825	-825	-825	-825	-825
10	0	825	825	825	825	825	825	825	825	825
9	0	2454	2454	2454	2454	2454	2454	2454	2454	2454
8	0	4016	4016	4016	4016	4016	4016	4016	4016	4016
7	0	5469	5469	5469	5469	5469	5469	5469	5469	5469
6	0	6772	6772	6772	6772	6772	6772	6772	6772	6772
5	0	7891	7891	7891	7891	7891	7891	7891	7891	7891
4	0	8794	8794	8794	8794	8794	8794	8794	8794	8794
3	0	9458	9458	9458	9458	9458	9458	9458	9458	9458
2	0	4931	4931	4931	4931	4931	4931	4931	4931	4931
1	0	0	0	0	0	0	0	0	0	0

Fig. 11 The east–west momentum flux from wind forcing of the box model. It is the negative of the wind stress divided by the water density, expressed in units of $m^2\ s^{-2}$. It is displayed with the command CALL PRXY('WUSURF',TIME,WUSURF,IM,2,JM,1,1.0E-8).

FREE SURVACE

TIME = 5.0000 DAYS MULTIPLY ALL VALUES BY 1.000E-04

	1	3	5	7	9	11	13	15	17	19
20	0	0	0	0	0	0	0	0	0	0
19	0	-227	-216	-212	-210	-210	-208	-206	-204	-189
18	0	-161	-131	-123	-124	-128	-137	-152	-172	-182
17	0	-96	-44	-33	-35	-45	-61	-86	-121	-162
16	0	-46	25	39	35	21	-2	-37	-88	-153
15	0	-3	86	104	98	80	50	5	-58	-146
14	0	32	137	158	150	128	93	41	-34	-141
13	0	60	176	199	190	166	127	68	-16	-138
12	0	79	203	226	217	190	149	86	-4	-137
11	0	90	216	240	231	203	159	94	1	-137
10	0	91	216	240	230	203	158	94	0	-139
9	0	84	204	227	217	189	147	84	-7	-142
8	0	69	179	200	191	165	125	65	-19	-146
7	0	45	143	162	154	130	93	39	-37	-152
6	0	15	97	114	106	86	53	6	-60	-160
5	0	-21	43	56	50	33	6	-33	-88	-168
4	0	-63	-16	-7	-12	-24	-46	-76	-118	-179
3	0	-123	-94	-88	-91	-100	-114	-136	-164	-199
2	0	-164	-160	-164	-169	-175	-183	-194	-208	-221
1	0	0	0	0	0	0	0	0	0	0

Fig. 12 The box model sea level in meters after 5 days, displayed by CALL PRXY('FREE SURFACE',TIME,ELB,IM,2,JM,1,1,E-4).

AVERAGE U COMP.

TIME = 5.0000 DAYS MULTIPLY ALL VALUES BY 1.000E-04

	1	3	5	7	9	11	13	15	17	19
20	0	0	0	0	0	0	0	0	0	0
19	0	17	74	89	86	76	60	36	3	-28
18	0	106	172	186	184	174	158	135	103	57
17	0	93	159	173	171	161	147	125	94	45
16	0	80	141	155	153	144	131	111	82	36
15	0	67	121	132	131	124	112	95	69	28
14	0	53	97	107	105	100	90	76	55	22
13	0	39	71	78	77	73	66	56	40	15
12	0	23	43	47	47	44	40	34	24	9
11	0	7	14	15	15	15	13	11	8	3
10	0	-8	-15	-16	-16	-15	-13	-11	-8	-3
9	0	-24	-44	-48	-47	-45	-40	-34	-24	-9
8	0	-39	-72	-78	-77	-73	-66	-55	-40	-15
7	0	-54	-98	-107	-106	-100	-90	-76	-55	-22
6	0	-68	-121	-133	-131	-124	-112	-95	-69	-28
5	0	-80	-142	-155	-153	-145	-131	-111	-82	-36
4	0	-93	-159	-173	-171	-162	-147	-125	-93	-45
3	0	-106	-171	-186	-184	-175	-159	-136	-103	-57
2	0	-15	-73	-88	-86	-76	-60	-36	-3	29
1	0	0	0	0	0	0	0	0	0	0

Fig. 13 The box model barotropic east-west current in $m\ s^{-1}$ after 5 days, displayed by CALL PRXY('AVERAGE U COMP.',TIME,UAB,IM,2,JM,1,1.E-4).

AVERAGE V COMP.

TIME = 5.0000 DAYS MULTIPLY ALL VALUES BY 1.000E-04

	1	3	5	7	9	11	13	15	17	19
20	0	0	0	0	0	0	0	0	0	0
19	0	17	5	0	-2	-3	-5	-7	-9	14
18	0	39	10	0	-4	-7	-10	-15	-19	-14
17	0	61	15	0	-6	-10	-15	-22	-30	-36
16	0	81	20	0	-8	-13	-20	-29	-40	-54
15	0	99	24	0	-9	-16	-24	-34	-49	-69
14	0	113	28	0	-10	-18	-27	-39	-56	-80
13	0	124	30	0	-11	-19	-29	-43	-61	-88
12	0	130	32	0	-12	-20	-31	-45	-65	-92
11	0	133	33	0	-12	-21	-31	-45	-66	-94
10	0	130	32	0	-12	-20	-30	-45	-65	-92
9	0	124	31	0	-11	-19	-29	-43	-61	-88
8	0	113	28	0	-10	-18	-27	-39	-56	-80
7	0	99	25	0	-9	-16	-24	-34	-49	-69
6	0	81	20	0	-8	-13	-20	-29	-40	-54
5	0	60	16	0	-6	-10	-15	-22	-30	-36
4	0	39	10	0	-4	-7	-10	-15	-19	-14
3	0	17	5	0	-2	-3	-5	-7	-9	14
2	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0

Fig. 14 The box model barotropic north-south current in $m\ s^{-1}$ after 5 days, displayed by CALL PRXY('AVERAGE V COMP.',TIME,VAB,IM,2,JM,1,1.E-4).

DISTRIBUTION LIST

1. Office of Naval Research
   Code 1242 (10 copies)
   800 North Quincy Street
   Arlington, VA 22217-5000

2. Director, Atmospheric Sciences Directorate
   NOARL West (Code 400)
   Monterey, CA 93943-5000

3. Commanding Officer
   Fleet Numerical Oceanography Office
   Monterey, CA 93943-5000

4. Commanding Officer
   Naval Oceanographic Office
   Stennis Space Center, MS 39529

5. Technical Director
   CNOC (Code OOT)
   Stennis Space Center, MS 39529

6. Commanding Officer
   NOARL (Code 100)
   Stennis Space Center, MS 39529

7. Technical Director
   NOARL (Code 110)
   Stennis Space Center, MS 39529

8. Director, Ocean Sciences Directorate
   NOARL (Code 300)
   Stennis Space Center, MS 39529

9. Head, Ocean Sensing and Prediction Division
   NOARL (Code 320)
   Stennis Space Center, MS 39529

10. Library (3 copies)
    NOARL (Code 125)
    Stennis Space Center, MS 39529

11. Dr. J. Dana Thompson
    NOARL, Code 320
    Building 1103, Room 248
    Stennis Space Center, MS 39529

12. Dr. William J. Schmitz, Jr.
    Department of Oceanography
    Woods Hole Oceanographic Inst.
    Woods Hole, MA 02543

13. Dr. William Holland
    National Center for Atmospheric Research
    P. O. Box 3000
    Boulder, CO 80307

14. UCAR Library
    P. O. Box 3000
    Boulder, CO 80307

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. Agency Use Only *(Leave blank).*	2. Report Date.	3. Report Type and Dates Covered.
	March 1991	Special Project

4. Title and Subtitle.

A User's Manual for the Princeton Numerical Ocean Model

5. Funding Numbers.

Program Element No. 0601153N
Project No. R310300
Task No. 801
Accession No. DN250022

6. Author(s).

William P. O'Connor

7. Performing Organization Name(s) and Address(es).

Institute for Naval Oceanography
Building 1103, Room 233
Stennis Space Center, MS 39529-5005

8. Performing Organization Report Number.

SP-5

9. Sponsoring/Monitoring Agency Name(s) and Address(es).

Naval Oceanographic and Atmospheric Research Laboratory
Code 110, Stennis Space Center, MS 39529

10. Sponsoring/Monitoring Agency Report Number.

11. Supplementary Notes.

12a. Distribution/Availability Statement.

Approved for public release; distribution is unlimited.

12b. Distribution Code.

13. Abstract *(Maximum 200 words).*

This is a user's guide for the numerical ocean model developed at Princeton University and presently in use at the Institute for Naval Oceanography. The basic model has been applied successfully to study such diverse regions as Chesapeake Bay (Blumberg 1977), the South Atlantic Bight (Blumberg and Mellor 1983), the Mid Atlantic Bight (Blumberg and Kantha 1985), the Gulf of Mexico (Blumberg and Mellor 1985), New York Harbor (Oey et al 1985a,b,c), Delaware Bay (Galperin and Mellor 1990a,b), and the western Atlantic including the Gulf Stream (Mellor and Ezer 1991). It can model regions from the size of bays and estuaries to basin scale North Atlantic domains. A good discussion of modeling bays and coastal oceans is given by Blumberg and Oey (1985).

14. Subject Terms.

(U) INO (U) NAOPS (U) PRIDER (U) PEDAM (U) OGCM

15. Number of Pages. 69

16. Price Code.

17. Security Classification of Report.	18. Security Classification of This Page.	19. Security Classification of Abstract.	20. Limitation of Abstract.
Unclassified	Unclassified	Unclassified	SAR